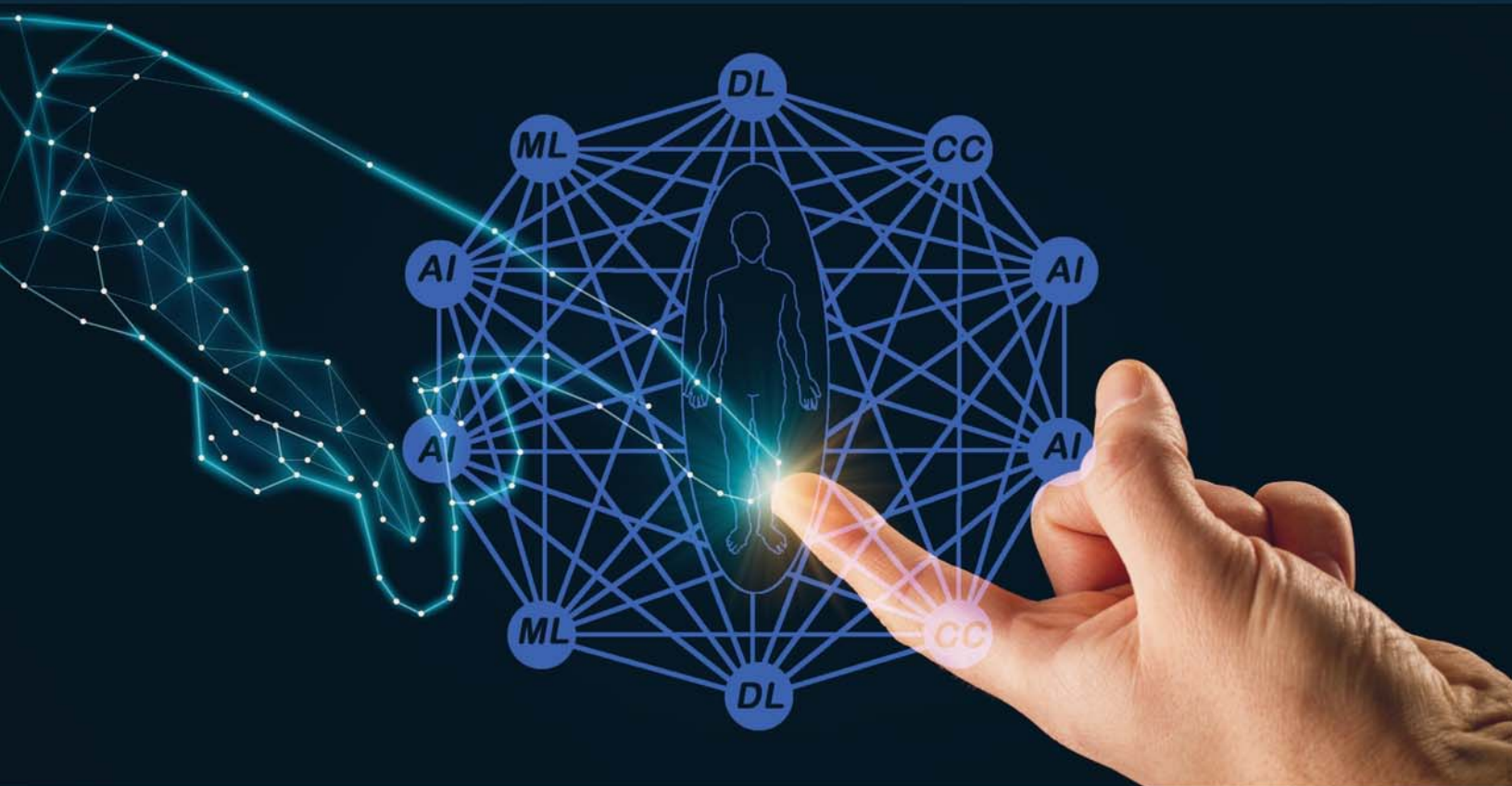


Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models



Jorge Garza-Ulloa



Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models

This page intentionally left blank

Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models

Jorge Garza-Ulloa

CEO/Director Garza Ulloa Research Consulting Services,
University of Texas, El Paso, TX, United States



ACADEMIC PRESS

An imprint of Elsevier

Academic Press is an imprint of Elsevier
125 London Wall, London EC2Y 5AS, United Kingdom
525 B Street, Suite 1650, San Diego, CA 92101, United States
50 Hampshire Street, 5th Floor, Cambridge, MA 02139, United States
The Boulevard, Langford Lane, Kidlington, Oxford OX5 1GB, United Kingdom

Copyright © 2022 Elsevier Inc. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or any information storage and retrieval system, without permission in writing from the publisher. Details on how to seek permission, further information about the Publisher's permissions policies and our arrangements with organizations such as the Copyright Clearance Center and the Copyright Licensing Agency, can be found at our website: www.elsevier.com/permissions.

This book and the individual contributions contained in it are protected under copyright by the Publisher (other than as may be noted herein).

Notices

Knowledge and best practice in this field are constantly changing. As new research and experience broaden our understanding, changes in research methods, professional practices, or medical treatment may become necessary.

Practitioners and researchers must always rely on their own experience and knowledge in evaluating and using any information, methods, compounds, or experiments described herein. In using such information or methods they should be mindful of their own safety and the safety of others, including parties for whom they have a professional responsibility.

To the fullest extent of the law, neither the Publisher nor the authors, contributors, or editors, assume any liability for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, or from any use or operation of any methods, products, instructions, or ideas contained in the material herein.

British Library Cataloguing-in-Publication Data

A catalogue record for this book is available from the British Library

Library of Congress Cataloging-in-Publication Data

A catalog record for this book is available from the Library of Congress

ISBN: 978-0-12-820718-5

For information on all Academic Press publications
visit our website at <https://www.elsevier.com/books-and-journals>

Publisher: Mara Conner
Acquisitions Editor: Chris Katsaropoulos
Editorial Project Manager: Emily Thomson
Production Project Manager: Niranjan Bhaskaran
Cover Designer: Greg Harris

Typeset by MPS Limited, Chennai, India



Dedication

*Every human is unique, and their health and well-being must be assured with self-care under the supervision of **healthcare professionals** who now have easy access to use **Biomedical engineering tools based on artificial intelligence models** for better understanding of day-to-day pattern changes in their patients, and obtain measurable values for more precise feedback and recommend appropriate advice and support for staying healthy.*

This book is dedicated to all those who face *human illness, diseases, and disorders*. These three can be defined as follows:

- *Human illness* is defined as body damage that needs to be cured such as infections, injuries, cells degeneration.
- *Human diseases* can be defined as states or reactions that must be managed, for example, pain, discomfort, weakness, and fatigue, and
- *Human disorders* can be defined as function abnormalities that must be treated such as physical, mental, genetic, emotional/behavioral, and functionals.

The complexity for the kind of analysis needed is bondless and can only be analyzed through the combination of multidisciplinary sciences such as *Biomedical Engineering, Cognitive Science, and Computer science* and applying different *Artificial Intelligent and cognitive models*.

*Helping to find better solutions that affect human health,
so that all can have a better, more productively, and enjoyable world!*

Jorge Garza-Ulloa

This page intentionally left blank

Contents

About the author	xi	1.6.4 Cognitive computing applying AI technologies	29
Foreword	xiii	1.6.5 Cognitive model obtainment	30
Preface	xv	1.6.6 Inference engine	31
Acknowledgment	xix	1.7 Neuroscience, cognitive science, and AI models	31
		1.7.1 The near future of neuroscience, cognitive science, and AI-ML-DL-CC	34
1. Biomedical engineering and the evolution of artificial intelligence	1	References	35
1.1 Introduction	1		
1.2 Biomedical engineering	2	2. Introduction to cognitive science, cognitive computing, and human cognitive relation to help in the solution of artificial intelligence biomedical engineering problems	39
1.2.1 Main purposes of AI in biomedical engineering	2	2.1 Introduction	39
1.2.2 AI and biomedical engineering help in medical education	3	2.2 Brain, spinal cord, and nerves	40
1.3 Artificial intelligence	4	2.3 Neurons and neural pathways in cognition	41
1.3.1 Turing test/Turing machine	4	2.3.1 Neurons and cognition	41
1.3.2 Basic types of AI systems based on capabilities	5	2.3.2 Neural pathway and cognition	43
1.3.3 Basic types of AI systems based on functionality	7	2.3.3 Dopamine pathways and cognition	44
1.3.4 AI technology evolution	8	2.4 Cognitive science	45
1.3.5 AI in industries	10	2.5 Natural Language Processing	48
1.4 Machine learning	11	2.5.1 Step 1) Reading dataset for NLP	49
1.4.1 ML seven specific steps	11	2.5.2 Step 2) Preprocessing text for NLP	49
1.5 Deep learning	16	2.5.3 Step 3) Data vectorization in NLP	49
1.5.1 Difference between deep learning and machine learning	16	2.5.4 Step 4) Feature engineering in NLP	50
1.5.2 Types of artificial neural networks	17	2.5.5 Step 5) ML model selection for NLP	50
1.5.3 Feed forward neural network	18	2.6 MATLAB [®] toolboxes solution for natural language processing	50
1.5.4 Backpropagation neural network	20	2.6.1 Natural Language Processing applications with MATLAB	51
1.5.5 Recurrent neural networks	22	2.6.2 "NLP Topic Models" with MATLAB	52
1.5.6 Memory augmented neural networks	25	2.6.3 "NLP audio files" with MATLAB	57
1.5.7 Modular neural networks	25	2.6.4 "NLP Text to Speech" using MATLAB	58
1.5.8 Evolutionary deep neural networks	26	2.6.5 "NLP Speech to Text" with MATLAB and IBM Cloud API	60
1.6 Cognitive science	27		
1.6.1 Cognitive computing	28		
1.6.2 Signal recognition instruments	29		
1.6.3 Cognitive detection of human-like abilities	29		

2.7 Cloud service and AI	76	4.2.4 Survival models	178
2.7.1 Cloud service providers and AI	76	4.2.5 Association Rules	178
2.8 IBM Cloud, IBM Watson, and Cognitive apps	77	4.3 ML clusters, classification, and regression models	179
2.8.1 IBM Cloud solution for natural language processing	78	4.4 Naive Bayes family models for supervised learning	180
2.8.2 IBM Cloud exercise to create APIs for NLP applications	78	4.4.1 Models: Gaussian Naive Bayes, multinomial Naive Bayes, Bernoulli Naive Bayes, Kernel Naive Bayes	181
2.9 The future of the relationship between cognitive science, cognitive computing, and human cognition	110	4.5 k-Nearest neighbor family models for supervised learning	181
References	110	4.5.1 Family models: fine kNN, medium kNN, coarse kNN, cosine kNN, cubic kNN, and weighted kNN	181
Further reading	111	4.6 Decision trees family models for supervised learning	182
4.6.1 Family models: fine decision tree, medium decision tree, and coarse decision tree		4.7 Support vector machine family members	183
3. Artificial intelligence models applied to biomedical engineering	113	4.7.1 Family models: linear SVM, fine Gaussian SVM, medium Gaussian SVM, coarse Gaussian SVM, quadratic SVM, and cubic SVM	183
3.1 Introduction artificial intelligence and biomedical engineering	113	4.8 Artificial neural network family models	184
3.2 AI optimization in biomedical engineering	114	4.8.1 Feed forward neural network family models: perceptron, multilayer perceptron, radial basis network, probabilistic neural network, extreme learning machine	184
3.3 Evolutionary algorithms for AI optimization in BME	115	4.8.2 Backpropagation neural networks	185
3.3.1 A typical evolutionary algorithm	116	4.9 Discriminant analysis family models	185
3.3.2 Genetic algorithms for AI optimization in BME	118	4.9.1 Family models: linear discriminant analysis, quadratic discriminant analysis	185
3.3.3 Genetic algorithm for AI optimization in BME under MATLAB®	120	4.10 Logistic regression classifier	186
3.3.4 General analysis and optimization of 2D and 3D data in biomedical engineering	121	4.10.1 Family models "logistic regression"	186
3.3.5 MATLAB analysis and optimization of "2D" data in biomedical engineering	124	4.11 Ensemble classifiers family models	186
3.3.6 MATLAB analysis and optimization of 3D data in biomedical engineering	126	4.11.1 Models: AdaBoost, RUSBoost, Subspace kNN, Random Forrest, Subspace discriminant	186
3.4 IBM watson studio for artificial intelligence	134	4.12 IBM ML Solution: IBM Watson SPSS	187
3.4.1 IBM SPSS Modeler Flow	136	4.12.1 SPSS Modeler flows > Modeling	187
3.4.2 IBM Watson using SPSS Modeler Flow for general dataset analysis	138	4.12.2 SPSS Modeler flows > Output	189
3.5 Examples of applications of evolutionary algorithms with other AI tools in biomedical engineering	170	References	333
References	172	Further reading	334
4. Machine learning models applied to biomedical engineering	175	5. Deep learning models principles applied to biomedical engineering	335
4.1 Introduction	175	5.1 Deep learning based on artificial neural networks	335
4.2 Choosing the best ML model	175	5.2 Feed forward neural networks types	336
4.2.1 Unsupervised learning	176		
4.2.2 Supervised learning	176		
4.2.3 Reinforcement learning	177		

5.2.1	Perceptron (P) or single-layer perceptron network	337	6.2.1	Recurrent Neural Network vanilla	509
5.2.2	Multilayer perceptron	338	6.2.2	Long/short-term memory	511
5.2.3	Radial basis function network	340	6.2.3	Gated recurrent unit networks	512
5.2.4	Probabilistic neural network	342	6.2.4	Recurrent convolutional neural networks	536
5.2.5	Extreme Learning Machine	343	6.2.5	Regional-Convolutional Neural Network Object detection in AI models	536
5.3	Shallow neural network	346	6.2.6	Hopfield Network	554
5.3.1	Research 5.1 Feed Forward Neural Network to Analyze "Human Body Fat"	346	6.2.7	Boltzmann Machine	561
5.3.2	Research 5.2 Neural Network for clustering based on "Self-Organizing MAP through a Shallow Neural Network" to analyze Surface Electromyography signals	370	6.2.8	Restricted Boltzmann Machine	562
5.3.3	Research 5.3 Neural Network for Dynamic Time series based on a "NARX is a nonlinear autoregressive exogenous model" to analyze vertical Ground Reaction Forces signals	389	6.2.9	Liquid State Machine	569
5.4	Backpropagation neural networks types	411	6.2.10	Echo State Network	570
5.4.1	Auto Encoder	411	6.2.11	Korhonen Network	580
5.4.2	Variational Auto Encoder	413	6.3	Memory augmented neural networks types	581
5.4.3	Denoising Auto Encoder	414	6.3.1	Neural Turing machine	581
5.4.4	Sparse Auto Encoder and stacked auto encoders	415	6.3.2	Differentiable Neural Computers	583
5.4.5	Deep Convolution Network or ConvNet	437	6.4	Modular Neural Networks types	592
5.4.6	Deconvolutional network	438	6.4.1	Deep Belief Network	592
5.4.7	Deep Convolutional Inverse Graphics Network	439	6.5	Evolutionary Deep Neural Networks types	601
5.4.8	Generative Adversarial Network	439	6.5.1	Capsule Networks	601
5.4.9	Deep Residual Network or Deep ResNet	440	6.5.2	Attention networks	603
5.5	Transfer learning from pretrained deep learning networks	442	References	604	
5.5.1	Research 5.5 "Pretrained Deep Convolutional Neural Network to obtain an AI model to classify Mammograms standard views types"	443	Further reading	607	
5.5.2	Research 5.6 modify a "Pretrained Deep Convolutional Neural Network" to obtain an AI model to "classify Mammograms view type and suggest breast abnormalities as possible breast tumor"	461	7.	Cognitive learning and reasoning models applied to biomedical engineering	609
5.5.3	Research 5.7 "custom Deep Convolutional Neural Network" to obtain an AI model to "classify Cervical X-rays view types"	482	7.1	Introduction	609
References	505	7.2	Artificial intelligence and Cognitive Computing Agents System (AI-CCAS)	609	
6.	Deep learning models evolution applied to biomedical engineering	509	7.3	Inference engine and research example	612
6.1	Deep learning models evolution	509	7.3.1	Research 7.1	612
6.2	Recurrent neural networks types	509	7.3.2	Research 7.2	615
			7.4	Action generation	627
			7.5	Business intelligence in healthcare	627
			7.6	Learning and reasoning relationship of biomedical engineering, cognitive science, and computer science through artificial intelligent models	629
			7.6.1	Cognitive learning and reasoning	629
			7.6.2	Deductive reasoning	630
			7.6.3	Inductive reasoning	630
			7.6.4	Abductive reasoning	631
			7.6.5	Abductive reasoning for medical diagnosis	634
			7.6.6	Metaphoric reasoning	635
			7.6.7	Neuro-Fuzzy logic reasoning as cognitive reasoning	636
			7.6.8	Visuospatial relational reasoning	639
			7.6.9	Cognitive learning and relationship with neuroscience of reasoning	640

7.7 Cognitive Learning and Reasoning research example applying AI-CCAS framework	640		
7.7.1 Research 7.3	641		
7.7.2 Research 7.4	653		
7.7.3 Research 7.5	661		
7.8 Challenge research for “Applied Biomedical Engineering using Artificial Intelligence and Cognitive Models”	667		
7.8.1 Challenge research project # 1: “Inductive Reasoning AI evaluation test for neurologic diseases patients under Cognitive Learning and Reasoning applying Cognitive Computing”	667		
7.8.2 Challenge research project # 2: “Abducting Reasoning using AI evaluation tests for patients under Cognitive Learning and Reasoning applying Cognitive Computing”	669		
7.8.3 Challenge research project # 3: “Metaphoric reasoning for clinical diagnosis using Cognitive Learning and Reasoning applying Cognitive Computing”	669		
7.8.4 Challenge research project # 4: “Neurologic – evaluating anxiety in neurologic diseases using Cognitive Therapy Theory using Cognitive Learning			
			and Reasoning with Cognitive Computing”
			670
		7.8.5 Challenge research project # 5: Analyze Neurologic opinion words with positive and negative frequently used to describe patient’s behavior with the symptoms labeled”	671
		7.8.6 Challenge research project # 6: “Classify status of neurologic disease patients analyzing their images, movements in real time video, and speech	672
		7.8.7 Challenge research project # 7: “human voice cognitive analysis for cognitive services as voice therapy”	672
		7.8.8 Challenge research project # 8: Cognitive Behavioral Therapy	673
		7.8.9 Challenge research project # 9: detection of “prefatigue/fatigue by stress and anxiety”	673
		7.8.10 Top Challenge research project # 10 building a Cognitive health Dashboard	674
		References	674
		Further Reading	676
		Index	677

About the author



Jorge Garza-Ulloa, CEO/Director of Garzaulloa Research Consulting Services at <http://www.garzaulloa.org/>. He graduated from the doctoral program in Electrical & Computer Engineering at the University of Texas at El Paso (UTEP) in May 2013. He received his Master of Science degree in electrical and computer engineering in 1980 from the University of Massachusetts at Amherst. Dr. Garza-Ulloa has been the recipient of numerous honors and awards including a University of Texas at El Paso Graduate School Research Award, Research Schellenberg Foundation, Stern Foundation, Elsevier grants, and others. Dr. Garza-Ulloa is currently teaching at the University of Texas at El Paso, United States and continues his Biomedical Engineering research at Research Consultant Services at El Paso, Texas, United States.

Dr. Garza-Ulloa is the author of the textbook/research books:

- *Applied Biomedical Engineering Using AI and Cognitive Models* (Elsevier 2021), which focuses on the relationships between three different multidiscipline engineering branches: Biomedical Engineering, Cognitive Science and Computer science through Artificial Intelligent models to study how the mental process of the information during cognition when “human illness, diseases, and disorders,” especially neurologic diseases, are present in the human body and

- *Applied Biomechanics Using Mathematical Models* (Elsevier 2018) with the goal of developing biomedical solutions for degenerative disease and injuries, that affect the human movements based on the mathematical modeling of human kinematics and kinetics to analyze motor-symptoms.

Dr. Garza-Ulloa has published his research in international journals including

- *Update on Parkinson’s Disease* at American Journal of Biomedical Science & Research (2019);
- *Theory of Stress-Anxiety-Sleep Disorders-Neural Damage Cyclic Chain and the Progression of Parkinson’s Disease* at the American Journal of Biomedical Science & Research (2019);
- *A Parkinson’s disease journey from patient side view* at American Journal of Biomedical Science & Research (2019);
- *Assessment and evaluation of the dynamic behavior of muscles with special reference to subjects with Diabetes Mellitus*, Dr. Garza-Ulloa dissertation supervised by Dr. Thompson Sarkodie-Gyan (2013);
- *A Novel Mathematical Model for the Validation of the Ground Reaction Force Sensor in Human Gait Analysis* at Journal Measurement Elsevier (2012);
- *A novelty mathematical model to predict Transition-to-Fatigue during isometric exercise on muscles of the lower extremities* at Engineering Supplement: World Congress on Engineering and Technology (2012);
- *The Development/Design Theory and Methodology Model for Mechatronics* at International conference meeting: ASEE Annual Conference (2012); and others.

He has founded three international technologies research consulting companies and has been a leader in developing new specialized solutions in multidisciplinary fields.

This page intentionally left blank

Foreword

This book focuses on the relationships between three different multidisciplinary engineering branches: Biomedical Engineering, Cognitive Science, and Computer Science through different Artificial Intelligence models in order to analyze human illness, diseases, and disorders, with special emphasis on the mental processes of the information

during cognition when disorders of neurologic diseases are present in the human body, with the purpose of evaluating their nonmotor symptoms that will help find solutions for treatments, follow-ups, and, as a consequence, improve their quality of life.

Jorge Garza-Ulloa

This page intentionally left blank

Preface

This book *Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models* aims to help people understand the importance of applying artificial intelligence (AI) and cognitive models following the development of mathematical models to be applied to many illnesses, diseases, and injuries with a special focus on nonmotor aspects of neurologic disease process using a biomedical engineering dataset to be analyzed, processed, classified, and predicted and obtain many AI models to help in the developing of solutions that can be implemented in a standalone way or in revolutionaries new frameworks as the *Proposed General Architecture framework of a Cognitive Computing Agents System (AI-CCAS)* with special emphasis in their relationship with *neuroscience of reasoning* based on *Cognitive Learning and Reasoning (CL&R) using Cognitive Computing (CC)*.

This book has seven chapters, covering the following topics:

Chapter 1, Biomedical Engineering and the Evolution of Artificial Intelligence

A study of the interactions on different injuries, illnesses, and diseases with special emphasis in Neurology with Cognitive Science in Biomedical Engineering solutions based on the evolution of AI through machine learning (ML), deep learning (DL), and CC. Provides an introduction to the general framework architecture for AI-CC Agent Systems (AI-CCAS) to help in the detection of cognitive human-like abilities with the objective of developing AI methods for medicine and healthcare through the analysis of numeric data, images, speech, and text to help in the detection and diagnosis of illness or determine health conditions, with special emphasis on neurologic diseases.

Chapter 2, Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to Help in the solution of AI Biomedical Engineering Problems

Provides an introduction for the analysis of bodily injuries, diseases, and neurological disorders separated

basically as motor symptoms (related to movement disorders) and nonmotor symptoms (related to cognition and not related to movement disorders). “Human Cognitive Developmental Stages” and their relation to neurons and neural pathways. These include Cognition and its integration with multidiscipline sciences, Natural Language Processing applications, NLP Text to speech, NLP Speech to Text, Audio Labeler for ML, and NLP analysis for Sentiment, Emotion, Keywords, Entities, Categories, Concept and Semantic Roles with MATLAB® and API as a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other services through IBM Cloud services.

Some examples and exercises for NLP Topics are in this chapter:

- *Theoretical example of Bag of words NLP method* (see example 2.1)
- *NLP Topic Models between a blog with a patient with neurologic disease and a researcher* (see Research 2.1)
- *NLP audio files with MATLAB* (see example in Section 2.6.3)
- *NLP Text to Speech using MATLAB* (see example in Section 2.6.4)
- *NLP text to Speech as action generation using MATLAB* (see example in Research 2.3)
- *NLP text to Speech as action generation using MATLAB and IBM Cloud API* (see example in Research 2.4)
- *Creating more IBM Cloud API services and testing them using from command lines with curl as an open software* (see example at Research 2.5)
- And others

Chapter 3, Artificial Intelligence Models Applied to Biomedical Engineering

Provides an introduction for applying AI algorithms to resolve biomedical engineering problems through

evolutionary algorithms using the evolution of the species, trying to emulate the natural evolution as Genetic Algorithms, Swarm Algorithms, traditional search methods, optimization of numeric value problems in 2D and 3D, visual analysis of Biomedical Engineering datasets of different diseases from different Bioinstruments to analyze the relationship between their attributes and applying AI tools.

Examples and tutorials in MATLAB and IBM Watson Studio SPSS Modeler Flow as

- *Genetic Algorithm to deduct items in a bag from a list that recommends taking to the hospital when a patient will stay for a month* (see example in Research 3.1 and 3.2)
- *Analysis and optimization of 2D data values from Body measurement of nerve contractions* applying MATLAB (see example in Research 3.3)
- *Analysis and optimization of 3D data values from the center of mass of an upper extremity—right arm movement from a patient* applying MATLAB (see example in Research 3.4)
- *Diabetes analysis* using IBM Watson using SPSS Modeler Flow (see example in Research 3.5)
- And others

Chapter 4, Machine Learning Models Applied to Biomedical Engineering

ML is presented as a subset of AI following the steps to obtain a prediction model based on pattern recognition in data using clustering, classifiers, and regression models. Advice was given as to how to follow, select, find, and implement the best ML models, according to the type of ML problem. Generally, this can be unsupervised learning, supervised learning, reinforcement learning, survival models, association rules, and others. A study is presented of different the ML Models Families applying IBM Watson SPSS Modeler Flow/IBM Watson Machine Learning applications and MATLAB ML solution under the Statistics and Machine Learning Toolbox with research tutorial examples to analyses and obtain prediction models for different biomedical datasets as:

- *K-Means ML Model for Diabetes* using IBM Watson SPSS Modeler Flow (see example in Research 4.1)
- *Decision Tree ML Model for Heart disease* using IBM Watson SPSS Modeler Flow (see Research Tutorial 4.2)
- *Kidney disease ML Auto Classifiers Models* and deploy the best model using IBM Watson SPSS Modeler Flow (see Research Tutorial 4.3)
- *Breast cancer ML model* and deploy the best model using IBM Watson AutoAI experimenter (see Research Tutorial 4.4)
- And others

Chapter 5, Deep Learning Models Principles Applied to Biomedical Engineering

The underlying principle of DL is composed of neural networks inspired by the biological elements that form the human brain, as a collection of nodes emulating brain neurons, and their neuronal synapse connections as primary elements of a net, that combined form mid-level elements identified as artificial neural networks (ANNs), which in turn are combined with different architectures to form more complex networks. In this book, ANN is organized based on their architectural type, and the way of their different components are connected to define the specific learning goal as different types as Feed Forward Neural Network, Backpropagation Neural Networks, Recurrent Neural Networks, Memory Augmented Neural Networks, Modular Neural Networks, and Evolutionary Neural Networks.

The first two ANN types studied in this chapter are Perceptron (P), Multi-Layer Perceptron's (MLP), Radial Basis Function Network (RBF), Probabilistic Neural network (PNN), Extreme Learning Machine (ELM), Auto Encoders (AE), Variational Auto Encoder (VAE), Denoising Auto Encoder (DAE), Sparse Auto Encoder (SAE) & Stacked Auto Encoders, Deep Convolution Network (DCN), Deconvolutional Network (DN), Deep Convolutional Inverse Graphics Network (DCIGN), Generative Adversarial Network (GAN), Deep Residual Network (DRN), plus family nets under Shallow Neural Network, and the special kind of net learning known as Transfer Learning from Pretrained Deep Learning Networks.

All based on examples and practical research on Biomedical Engineering using existing *AI tools* from *MATLAB* and *IBM Watson Studio* with research tutorial examples to analyses and obtain prediction models for different biomedical datasets as:

- *Blood pressure reading using MLP*
- *Heart rhythm analysis using RFB*
- *Types of flu classification using PNN*
- *Diabetes value prediction using ELM*
- *Human body fat estimation under FFN*
- *Diabetes readings clustering for consecutives visits using Self-Organizing MAP (SOM)*
- *Ground reaction vertical forces using Neural Network for Dynamic Time series (NARX)*
- *X-rays images reconstructions using AE*
- *Classify Mammograms standard views types under Pretrained DCN*
- *Classify Mammograms view type and suggest breast abnormalities as possible breast tumor under modification of a pretrained DCN*
- *Classify Cervical X-rays view types using a custom DCN*
- And many others

Chapter 6, Deep Learning Models Evolution Applied to Biomedical Engineering

In this chapter, we focus on studying Deep Learning Models Evolution that combine mid-level elements with different connections. ANN forms more complex network family types organized as Recurrent Neural Networks, Memory Augmented Neural Networks, Modular Neural Networks and Evolutionary Neural Networks.

The ANN studied in this chapter are Vanilla Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Recurrent Convolutional Neural Networks (RCNN), Regional Convolutional Neural Network (R-CNN), Hopfield Network (HN), Boltzmann Machine (BM), Restricted Boltzmann Machine (RBM), Liquid State Machine (LSM), Echo State Network (ESN), Korhonen Network (KN) also known as the Self-Organizing Map (SOM), Neural Turing Machine (NTM), Differentiable Neural Computers (DNC), Deep Belief Network (DBN), Capsule Networks (CapsNet), Attention Network (AN), and others.

Many research examples are explained to be applied to Biomedical Engineering solutions as:

- *Classify videos based on human body movements for human falls detection using LSMT*
- *Classification for object detection of breast tumor in mammogram using R-CNN*
- *Reconstruct noisy chest X-rays images using HN*
- *Reconstruct noisy chest X-rays images using RBM*
- *Differentiate normal and pneumonia on chest X-rays using Reservoir Computing approach for a simulation of LSM based on node-neurons from Spiking Neural Networks (SNN)*
- *Analysis for COVID-19 respiratory transmission simulation droplets/mini-droplets emissions simulating an NTM based recursive function*
- *Analyze and differentiate normal and pneumonia chest X-rays using DBN*
- And many others

Chapter 7, Cognitive Learning and Reasoning Models Applied to Biomedical Engineering

In this chapter, we will focus on many prestudies and preanalyses of different Biomedical Engineering problems that need to be developed with specialized research projects applying the CL&R algorithm, that can be integrated into the Proposed General Architecture framework of a Cognitive Computing Agents System (AI-CCAS) with special emphasis of Cognitive Learning and its relationship with the neuroscience of reasoning using CL&R under CC. The complexity of the analysis needed is boundless, and only be analyzed through the multidisciplinary sciences, where interactions of science as biomedical engineering, neurology, cognitive sciences, and computer

science using tools with exponential technologies such as AI and others through its continuous exponential evolution that includes ML, DL, and CC. With the main purpose of obtaining useful AI models that can help analyze human health problems. Now is the time and place to apply them and many others in research challenge projects.

The AI cognitive models to be used for the AI-CCAS are studied in this chapter are: inference engine to extract the information needed from knowledge storage AI storage; Attention Network for NLP applying LSTM model to process information extracted by the AI-CCAS inference engine; CL&R using deductive reasoning, inductive reasoning, abductive reasoning, metaphoric reasoning, neuro-fuzzy logic reasoning, visuospatial relational reasoning, inferences fuzzy systems for fuzzy reasoning, cognitive sentiments analysis, reasoning evaluation for neurologic diseases, and others.

Many research examples are explained to be applied to Biomedical Engineering solutions for this chapter as:

- *Extract text information about COVID-19 (SARS-COV2) symptoms to develop the phenomenon fuzzy sets criteria to define the fuzzy input variables needed for the appropriate fuzzy rules in the inference engine*
- *Classify text of COVID-19 (SARS-COV2) symptoms extracted from the AI-CCAS inference engine using AN for NLP applying LSTM*
- *Abdominal body pain analysis to deduct cholecystitis applying inductive reasoning—causal arguments— inference of abduction*
- *Reasoning to be healthy applying abductive reasoning—causal arguments—backward chaining*
- *Deduct abnormal body temperatures applying abductive reasoning—causal arguments—paradigm case-based*
- *Reasoning to deduct if the patient has “flu” or “COVID-19” applying Abductive reasoning—Causal arguments—Generative coherence metric*
- *The impossible of reasoning for medical diagnosis of Parkinson’s disease applying Abductive Reasoning—causal arguments (cause \supset effect)*
- *Metaphor inference reasoning to find behavior similarities between cancer and COVID-19 by a metaphor*
- *Fuzzy Mamdani-type inference and/or Sugeno-type inference for a liquid medicine for control cough in patients*
- *Deductive reasoning evaluation for neurologic diseases patients using NLP under CC*
- *Cognitive sentiments analysis for neurologic diseases that affect mood changes applying NLP with CC applying CNN-NLP model*
- *Linguistic Neuro-Fuzzing Modeling to analyze breast cancer tumor*

- *Challenge research to develop applications for Applied Biomedical Engineering using AI and Cognitive Models learned in this book*
 - Challenge research project #1: *Inductive Reasoning AI evaluation test for neurologic diseases patients under CL&R applying CC*
 - Challenge research project #2: *Abducting Reasoning using AI evaluation tests for patients under CL&R applying CC*
 - Challenge research project #3: *Metaphoric reasoning for clinical diagnosis using CL&R applying CC*
 - Challenge research project #4: *Neurologic – evaluating anxiety in neurologic diseases using Cognitive Therapy Theory using CL&R with CC*
 - Challenge research project #5: *Analyze Neurologic opinion words with positive and negative frequently used to describe patient's behavior with the symptoms labeled*
 - Challenge research project #6: *Classify status of neurologic disease patients analyzing their images, movements in real-time video, and speech*
 - Challenge research project #7: *Human voice cognitive analysis for cognitive services as voice therapy*
 - Challenge research project #8: *Cognitive Behavioral Therapy applying Cognitive Learning and its relationship with neuroscience of reasoning proposed as CL&R*
 - Challenge research project #9: *Detection of "prefatigue/fatigue by stress and anxiety*
 - Top Challenge research project #10: *Building a Cognitive health Dashboard*
- Book companion website with MATLAB(R) /IBM Watson examples and dataset used in the book: <https://www.elsevier.com/books-and-journals/book-companion/9780128207185>.

Acknowledgment

I would like to acknowledge all the people in my life, especially those with “illness, diseases, and disorders,” that trust in health care and have trusted in me to describe their limitations and their suffering, along with those who have provided suggestions for the ideas and the solutions explained in this book.

I wish to acknowledge my family, who allowed me to concentrate on this in-depth research for 3 years to develop, organize, and write this textbook book; especially my wife, in the advanced stages of Parkinson’s disease. Being her 24/7 caregiver inspired me to connect her needs with the needs of many patients in order to reach an understanding of their feelings, and the urgency to find and develop processes to measure their status, to analyze the evolution of follow-ups, to develop analysis, and to try to find solutions for curing them or at least improving their quality of life. “Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models” can be used for learning and for analyzing the slow day-by-day cognitive deterioration, in addition to the other symptoms of degenerative diseases.

Special thanks go to all the people that helped in many ways in the realization of this book:

- Raul J Arizpe President and CEO,
- Diego Cruz IT, Ph.D UTEP Candidate IT System Director,
- Hugo Isuani. MD,FACR, Radiologist,
- Heidi Taboada-Jimenez, Ph.D., Associate Dean for Research & Graduate and Associate Professor, IMSE
- Jaime Sanchez, Ph.D, Lecturer, IMSE
- Vladik Kreinovich, Ph.D Professor, CS
- Thomas Boland, Ph.D Professor Director of Biomedical Engineering Program and many students, faculty & researchers at UTEP
- Pablo Rangel Ph.D. Assistant Professor S&E-School Of Engineering & CS
From University of Trás-os-Montes é Alto Douro
- Ivan Miguel Serrano Pires Ph.D Invited Assistant Professor, Researcher Instituto de Telecomunicações
From Medical Center of the Americas Foundation (MCA) & BIO El Paso-Juarez:

- Emma W. Schwartz, President and all MCA directives and Team members
From Technological Institute of Ciudad Juárez (Tecnológico Nacional de México Campus Cd. Juárez, (ITCJ):
- Mtro. Hermenegildo Lagarda Leyva Director ITCJ
- and students, researches and Faculty members
From Viomed Consulting LLC
- Jesus Carrillo CMDCP
From Mechatronics Automation
- Javier Acosta, Engineer, Director
- Charlie Yates, Ph.D President Chihuahua Cluster Energy & Consejo Nacional Clusters Energéticos of Mexico (CONACEM)
- Jorge Young F. Eng. President Metropolitan Energy Cluster of México
- Victor Hernandez, Engr. President Cluster of Artificial Intelligence & Cognitive Applications.
From Tesla Artificial Intelligence
- Marcos Barraza, Engr.
From The Piensa Institute
- Cruz A. Jimenez, Natural Intelligence & Business Intelligence
From Diest Consulting
- Hugo A. Becerra, President and many others researchers.

I really think that every time that we meet people is special, because every human being is unique, and we complement each other with our learnings and teaching in life. Each of us can help to improve life based on our expertise and we can work together for solutions to the problems, including health problems, that affect humanity. Thanks to *God*, to give us the opportunity to be *in the right place, with the right people, in the right time*, and to allow us to *help others*.

Jorge Garza-Ulloa

*CEO/Director Garza Ulloa Research Consulting Services,
University of Texas, El Paso, TX, United States
<http://www.garzaulloa.org>*

This page intentionally left blank

Biomedical engineering and the evolution of artificial intelligence

1.1 Introduction

This book, “*Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models*,” focuses on the relationship between three different multidisciplinary engineering branches: “*Biomedical Engineering*,” “*Cognitive Science*,” and “*Computer Science through Artificial Intelligent models*,” which includes “*Machine Learning*,” “*Deep Learning*,” and “*Cognitive Computing*,” to study how the nervous and musculoskeletal systems obey movement orders, with the goal of the understanding and obtainment of “*AI models*” of how information is mentally processed during “*cognition*” when injuries, illness, and/or neurologic diseases are present in the human body and affect the human body. Each of the multidisciplinary studies in this book is defined as the interaction between them as indicated in Fig. 1.1:

- *Biomedical engineering (BME)* as the application of engineering principles and design concepts to medicine and biology for healthcare purposes (e.g., analysis, diagnostic, confirmation, therapeutics).
- *Cognitive science (CoSi)* is the interdisciplinary scientific study of the mind and its processes. It examines

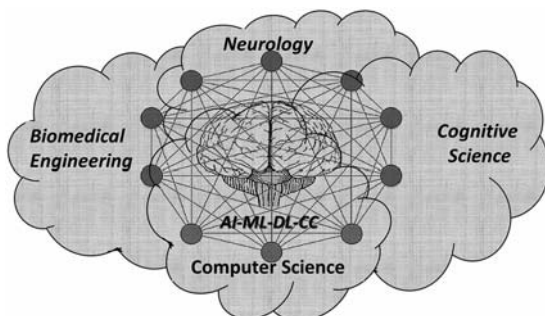


FIGURE 1.1 The interaction of Biomedical Engineering, Cognitive Science, and Computer Science, which includes Artificial Intelligence AI, Machine Learning ML, Deep Learning DL, and Cognitive Computing CC, can be used for analysis detection, classification, and forecast of neurologic diseases.

the nature, tasks, and the functions of human cognition as the process of acquiring knowledge and understanding through thought, experience, and the senses.

- *Computer science (CS)* is the scientific and practical approach to computation using algorithms as a self-contained sequence of actions to be performed as calculation, data processing, and automated reasoning. In this book the study of “*Artificial Intelligence (AI)*” is focused on the imitation of intelligent human behavior by a machine applying software algorithms to develop models based on “*Machine Learning*,” “*Deep Learning*,” and “*Cognitive Computing*.” Each is described as follows:
 - *Machine Learning (ML)* is a subset of “*AI*” that evolved from the study of pattern recognition and computational learning theory.
 - *Deep Learning (DL)* is another subset of “*AI*” and a special type of “*ML*” that simulates the neurons and synapses of the human brain in its approach to processing data.
 - *Cognitive Computing (CC)* is a subfield of “*AI*” that allow us to analyze human cognition to represent the process studied in “*Cognitive Science*.” “*Cognitive Technology or Cognitive computing*” helps humans take better decisions with the help of smart machines based on cognitive science that studies the human brain and how it functions. “*CC*” has also self-learning algorithms and language tools; they rely on “*data mining*,” “*pattern recognition*,” “*Natural Language Processing*” and others tools to collect information to feed themselves, and computer systems based on cognitive data, that can understand natural language and interact with humans in a more natural way, with the ability of these systems to understand, hold firmly, and reason the collected information.

The interactions of these three multidisciplines with “*Neurology*”—a branch of medicine that studies disorders and diseases of the nervous system including “*central*

nervous system (brain and spinal cord),” and “*peripheral nervous system*”—are studied in this book, with the main objective being the obtainment of “*AI models on Biomedical Engineering,*” especially with regard to injuries and neurologic diseases of the human body, and studying diseases of the brain, spine, and the nerves that connect them and the musculoskeletal system. Based on the fact that there are more than 600 diseases of the nervous system, such as brain tumors, epilepsy, Parkinson’s disease, and stroke, these diseases affect also the human “*cognitive system*” that sends orders from the “*central nervous system (CNS)*” through the “*peripheral nervous systems (PNS)*” to do tasks using the musculoskeletal system and the way of reasoning. These actions can be detected by many “*Bioinstruments (Biomedical Instruments)*” and “*cognitive data*” allowing us to apply “*AI using ML-DL-CC models**” through algorithms to analyze, detect, classify, and forecast the process of different illnesses and injuries of the human body.

Note:* Examples and exercises Artificial Intelligence using ML-DL-CC models are based on algorithms developed under MATLAB and IBM Watson Studio.

Important: The new concept for “*Cognitive Learning and its relationship with neuroscience of reasoning proposed as Cognitive Learning-Reasoning (CL&R) using Cognitive Computing (CC)*” is studied in [Chapter 7](#), Cognitive Learning and Reasoning Models Applied to Biomedical Engineering, with many research examples and challenges for research to resolve.

1.2 Biomedical engineering

“*Biomedical Engineering (BME)*” is defined as the application of engineering principles and design concepts to medicine and biology for healthcare purposes [1]. “*BME*” covers the gap between engineering and medicine, combining the design and problem-solving skills of engineering with medical and biological science to advance healthcare treatment, including diagnosis, monitoring, and therapy. “*BME*” has recently emerged as its own study, arising from many other engineering fields. Such an evolution is common from being an interdisciplinary specialization among already established fields, to being considered a field in itself.

Much of the work of “*BME*” consists of research and development, spanning a wide area of subfields. Prominent biomedical engineering applications include the development of biocompatible prostheses, many diagnostics, and therapeutic medical devices ranging from clinical equipment to microimplants, common imaging equipment such as “*magnetic resonance imaging (MRI)*” and “*Electroencephalography (EEG)*,”

regenerative tissue growth, pharmaceutical drugs, therapeutic biologicals, and many more that are developed every day. “*BME*” is a multidisciplinary field that has a lot of subfields such as “*Bioinformatics, Biomechanics, Biomaterials, Biomedical optics, Tissue engineering, Neural engineering, Pharmaceutical engineering, Clinical Engineering, Rehabilitation engineering, and Medical devices.*”

In all subfields of “*BME*” it is necessary to simulate the behavior of specific cases to understand them, and if it is possible to resolve these complex problems through the application of *Artificial Intelligence* models and achieve solutions for different purposes, such as confirm diagnoses, predict evolution, and many more.

Note:* For more information on Biomedical Engineering subfields, please read Chapter 1 of my book: “*Applied Biomechanics Using Mathematical Models*” [2].

1.2.1 Main purposes of AI in biomedical engineering

The purpose of “*AI in healthcare*” is the use of “*AI algorithms*” to approximate human cognition in the analysis of multiple medical data through “*AI models.*” Recent technological AI advances are transforming medical science and evolving biology through “*AI systems and AI applications of Biomedical Engineering*” with other multidisciplinary fields, such as healthcare, bioclinical, medical informatics, bioinformatics, medical data mining, medical systems using automated reasoning and Meta-reasoning, drug discovery, intelligent analysis of genomic and proteomic data, biomedical ontologies, medical imaging, and many other areas.

- “*Healthcare using artificial Intelligence*” has the objectives of finding and applying efficient algorithms for analyzing the relationship between prevention, treatment, and patient outcomes. AI helps healthcare in many different processes, such as management, clinical decision support systems, knowledge acquisition, and analysis of different illness and diseases, by applying “*AI algorithms.*”
- “*Bioclinical uses computational intelligence (CI)*,” where “*Bioclinical*” is a specialty in clinical trials that accelerates the development of new medical therapies, and “*CI*” is the theory, design application, and development of biological and linguistic paradigms using “*AI algorithms,*” encompassing computing paradigms like ambient intelligence, artificial life, cultural learning, artificial endocrine networks, social reasoning, and artificial hormone networks. Besides, “*CI*” plays a major role in developing successful intelligent

systems, including games and cognitive developmental systems [3].

- “*Medical informatics*” is the intersection of “*information science*,” “*computer science*,” and “*healthcare*.” This field deals with the resources, devices, and methods required to optimize the acquisition, storage, retrieval, and use of information in health and biomedicine.
- “*Bioinformatics*” is an interdisciplinary field combining “*biology*,” “*computer science*,” “*information engineering*,” “*mathematics and statistics*” to analyze and interpret biological data. “*Bioinformatics scientists*” design and apply the computer systems and databases used to organize and analyze large amounts of genomic, pharmacological, and other biological data.
- “*Medical data mining*” is the analysis of large datasets to discover patterns and use those patterns to forecast or predict the likelihood of future events, thus helping the healthcare industry to develop health systems to systematically use data and analytics to identify inefficiencies and best practices that improve care and reduce costs. The analytics can be “*descriptive (what happened)*,” “*predictive (what will happen)*,” and “*prescriptive (determine what to do)*” [4].
- “*Automated reasoning* and *Meta-reasoning*” are applied in “*biomedical engineering*.”
 - “*Automated reasoning*” is an area of “*computer science*,” “*cognitive science*,” and “*mathematical logic*” dedicated to understanding different aspects of reasoning. The study of automated reasoning helps to produce computer programs that allow computers to reason completely, or nearly completely, automatically.
 - “*Meta-reasoning is the reasoning about reasoning*” [5]. In a computer system, this means that the system is able to reason about its own operation. This is different from performing object-level reasoning, which refers in some way to entities external to the system. A system capable of meta-reasoning may be able to reflect, or introspect, that is, to shift from meta-reasoning to object-level reasoning and vice versa.
- “*Drug discovery*.” “*AI*” has the potential to stimulate and streamline drug discovery and development by increasing our understanding of complex biology, such as guide drug design, aggregate and synthesize information, understand mechanisms of disease, establish biomarkers, generate data and models, repurpose existing drugs, generate novel drug candidates, and other research phases.
- “*Intelligent analysis of genomic and proteomic data*.” “*Genomics*” can be broadly defined as the systematic study of genes, their functions, and their interactions. Analogously, “*proteomics*” is the study of proteins, protein complexes, their localization, their interactions, and posttranslational modifications [6].
- “*Biomedical ontologies*.” In the “*postgenomic era*,” “*biomedical ontologies*” are becoming increasingly popular in the computational biology community as the focus of biology has started to shift from mapping genomes to analyzing the vast amount of information resulting from functional genomics research. In fact, biomedical ontologies play a central role in integrating the information about various model organisms, acquired under different conditions, and stored in heterogeneous databases [7].
- “*Medical imaging*” is the technique and process of creating visual representations of the interior of a body for clinical analysis and medical intervention, as well as visual representation of the function of some organs or tissues (physiology). “*Medical imaging*” seeks to reveal internal structures hidden by the skin and bones, as well as to diagnose and treat disease. It is part of biological imaging and incorporates radiology, which uses the imaging technologies of X-ray radiography, magnetic resonance imaging, medical ultrasonography or ultrasound, endoscopy, elastography, tactile imaging, thermography, medical photography, and nuclear medicine functional imaging techniques, such as positron emission tomography (PET) and single-photon emission computed tomography (SPECT).
- Many others AI biomedical subfields.

1.2.2 AI and biomedical engineering help in medical education

Classically, a “*physician*” is defined as a professional who possesses special knowledge and skills derived from rigorous education, training, and experience [8], in other words “*medical education remains based on information acquisition and application*.” Currently, the amount of available medical knowledge now exceeds the organizing capacity of the human mind [9]. In addition, the skills required of practicing physicians will increase in two areas [10]: “*Collaborating with and managing Artificial Intelligence (AI) applications*, and *the need for more sophisticated mathematical understanding*.”

- “*Collaborating with and managing Artificial Intelligence (AI) applications*” that aggregate vast amounts of data, generate diagnostic and treatment recommendations, and assign confidence ratings to those recommendations [11]. Then, the long-standing approach of basing diagnostic or treatment choices on the “*average patient*” in a large population is no longer precise enough to meet the standards of personalized medicine. As a result, treatments for patients with different physical, cultural, and genetic attributes will vary in personalized medicine.

- “The need for more sophisticated mathematical understanding” is driven by the analytics of precision and personalized medicine generated by AI. The ability to correctly interpret probabilities requires mathematical sophistication in stochastic processes, something current medical curricula address inadequately.

Biomedical Engineering and AI can help to manage the medical information that grows constantly by developing more efficient algorithms, easy to use medical devices and medical applications with the goals of help physicians to:

- Focus on “knowledge capture, not knowledge retention.”
- Facilitate collaboration with and management of information.
- Enable simpler ways to understand probabilities and how to apply them for AI clinical decisions.

These AI systems and their applications are developed by applying many different algorithms, such as “Genetic algorithms,” “Bayesian models,” “Machine Learning,” “Deep Learning Artificial Neural Networks,” “Expert Systems,” “Fuzzy Logic,” and many more “AI methods” that are explained in this book, but a continuous learning attitude is recommended in order to try to stay up-to-date in these “exponential technologies.”

1.3 Artificial intelligence

“Artificial intelligence (AI)” is an area of computer science that emphasizes the creation of intelligent machines that work and react like the human brain. In other words, “AI” is any software process that enable machines to “mimic human intelligence” through computer systems. These processes include “learning,” “reasoning,” and “self-correction.”

- “Learning” is based on the acquisition of information and rules created for using the information.
- “Reasoning” is the application of rules to reach an approximation of a conclusion.
- “Self-correction” is achieved when the acquainted information grows, and the process adjusts the rules continually in a circular way of learning and reasoning to achieve a better approximation of the conclusion.

1.3.1 Turing test/Turing machine

The term “Artificial Intelligence” was created by John McCarthy in 1956 when he held the first academic conference on this subject [12]. “AI” was documented by Alan Turing in his paper: “Computing Machinery and Intelligence,” which opened the door to this field with the notion of machines being able to simulate human beings

and their ability to do intelligent things, such as play chess, analysis, classification, prediction, etc. Turing then went on to propose a method for evaluating whether machines can think, which came to be known as the “Turing Test,” as a central and long-term goal for “AI research.” Will we ever be able to build a computer that can sufficiently imitate a human to the point where a suspicious judge cannot tell the difference between human and machine?

Alan Turing worked on the problem to help define a system for identifying which statements could be proven. In the process, he proposed the “Turing Machine” concept, and in a research paper he defined a “computing machine” with the ability to read and write symbols to a tape using those symbols to execute an algorithm [13]. This paper and the “Turing Machine” provided the basis for the “theory of computation.” Initially it looked to be difficult but possible if hardware technology reached a certain point, only to reveal itself to be far more complicated than initially thought with progress slowing to the point where some wonder if it will ever be reached. Despite decades of research and great technological advances the “Turing test” still sets a goal that “AI researchers” strive toward while finding along the way how much further we are from realizing it.

“A complete AI system must think and act humanly, think and act rationally,” as indicated in Fig. 1.2. The AI system can be tested with the following steps:

- “Think humanly” can be reached by going inside the actual working of human minds through introspection, trying to catch our own thoughts, and psychological experiments.
- “Act humanly” can be tested using the “Turing Test” approach based on a smart system that possesses the following capabilities: “Natural Language Processing,” “Knowledge Representation,” “Automated Reasoning,” and “Machine Learning.”
 - “Natural Language Processing (NLP)” to enable it to communicate successfully in English and/or other human languages.
 - “Knowledge Representation” to store what it is learning, reasoning, sensing, seeing, or hearing.
 - “Automated Reasoning” is the ability to use the stored information to answer questions and draw new conclusions,
 - “Machine Learning (ML)” adapts to new circumstances and can detect and extrapolate patterns. “ML” can be defined as an application of “artificial intelligence (AI)” that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.
- Think rationally can be resolved by “the laws of thought” based on the three fundamental “laws of

logic”: “contradiction,” “exclude the middle,” and “the principle of identity” [14].

- “Law of contradiction” establishes that for all propositions “p,” it is impossible for both “p” and “not p” to be “true,” as shown in Eq. (1.1).

$$\text{Law of contradiction: } \sim(p \cdot \sim p),$$

in which \sim means “not” and “ \cdot ” means “logic AND function*”.

(1.1)

Note*: Logic AND function states that two or more events must occur together and at the same time for an output action to occur.

- “Law to exclude middle” establishes that either “p” or “ $\sim p$ ” must be “true,” there being no third or middle true proposition between them, as represented in Eq. (1.2).

$$\text{Law to exclude middle : } p \vee \sim p,$$

in which \vee means “Logical OR function*”

(1.2)

Note*: Logic OR function states that an output action will become TRUE if either one “OR” more events are TRUE, but the order at which they occur is unimportant as it does not affect the final result.

- “Law of principle of identity” asserts that a thing is identical with itself, as indicated in

$$\text{Law of principle of identity: } (\forall x)(x = x),$$

in which \forall means “for every”; or simply that “x is x”.

(1.3)

- “Act rationally” can be based on the “rational agent theory”; a rational agent can be anything that makes

decisions, typically a person, firm, machine, or software. “Rational agents” are also studied in the fields of “cognitive science (study of thought, learning, and mental organization),” ethics, and philosophy, including the philosophy of practical reason.

To study Artificial Intelligence, we must assume theoretically its main goal:
 “A complete AI system that must think and act humanly, think and act rationally, can be created and being tested.”

1.3.2 Basic types of AI systems based on capabilities

Today “AI” covers a broader area of “computer science” that makes systems or machines seem like they have human intelligence based on the following actions: when a machine can solve problems, complete a task, or exhibit other cognitive functions that humans can, then we refer to it as having “artificial intelligence.”

Actually, “AI” is something that we have to deal with every day in different application, such as e-mail communications, social media, web searching, stores and services, transportation, businesses, manufacturing, domestic appliances, and thousands more services and applications. We are in front of real “exponential AI technology” that is changing the world, and we must understand the advantages of using “AI and learn to differentiate between the possible types of AI based on capabilities,” which can be identified on three different levels as “Weak or Narrow AI,” “General AI,” and “strong AI,” as indicated in Fig. 1.2.

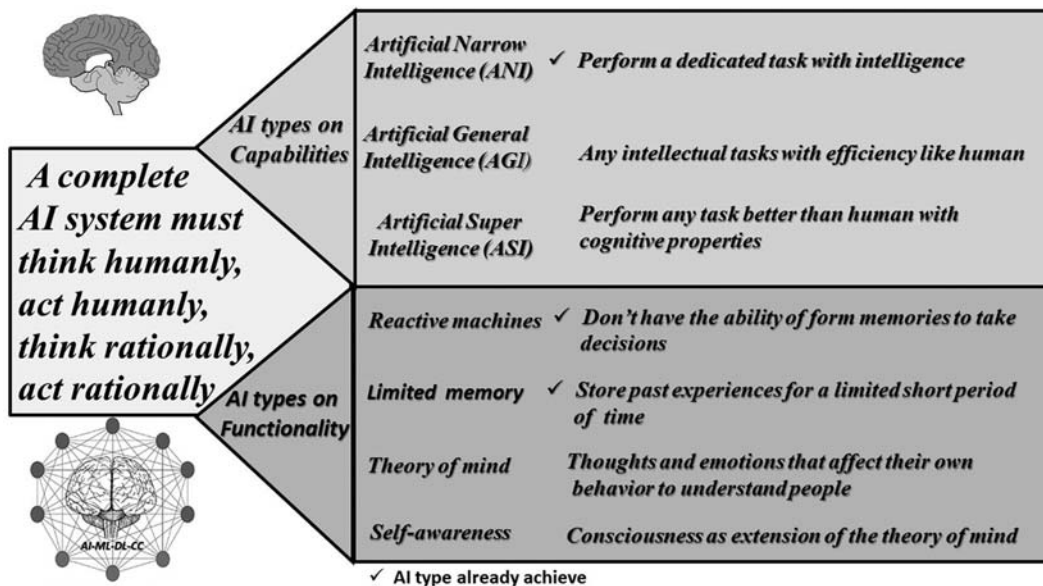


FIGURE 1.2 AI types based on Capabilities and Functionality.

- “*Weak or Narrow AI, also known as Artificial Narrow Intelligence (ANI).*” This type of AI enables a system to perform a dedicated task with intelligence, these days it is the most common available “*AI system.*” It is a kind of “*AI*” that cannot perform beyond its field or limitations, and it is only trained for one specific task and can fail in unpredictable ways if it goes beyond its limits, for example:
 - “*Virtual assistant or AI assistant are ANI applications*” based on computer programs that understand “*natural language voice commands and complete tasks,*” such as Apple Siri, Amazon Alexa, Google Assistant, Cortana and others. All of them process the human language then enter the information into a search engine and return to us with the results; they operate with a limited predefined range of functions. This can be tested when we make an abstract question like the meaning of life, which is why sometimes we get vague responses that often do not make any sense or we get the links to existing articles in the web.
 - “*IBM’s Watson supercomputer.*” This is another example of an *ANI* machine that uses an expert system approach combined with *Machine learning (ML)* and *Natural language processing (NLP)*.
 - *Self-driving cars or autonomous cars* are a special kind of “*ANI application*” made up of controls from “*multiple Narrow AI systems working together,*” to analyze in real time signals from different types of sensors, cameras, radar, and communications, thus enabling the vehicle to see, think, and make fast decisions.
 - Other examples are computer games (e.g., chess), purchasing suggestions on “*e-commerce,*” “*speech recognition,*” “*image recognition,*” and many more applications identified as “*ANI.*”
- “*General AI also known as Artificial General Intelligence (AGI)*” is a type of intelligence that performs any intellectual task with human-like efficiency; the idea is to have smarter systems that think like a human on their own. Currently, there is no such existing system that can perform any task as perfectly as a human; commonly they do only one *AI* specialized task without specialized human thinking and deduction functions. An “*AGI system*” will be not easy to replicate in machines using actual technology, because an “*AGI system*” must be based on the following *human facts*:
 - “*Humans have the ability to think abstractly, strategize, understand, and express beliefs*” or attitudes based on our thoughts and memories to make decisions or to come up with other creative ideas.
 - “*Humans are sentient creatures with the capacity to feel, perceive, or experience subjectively.*”
 - “*Humans have the abilities to plan, learn, reason, integrate prior knowledge, solve problems, make*

judgements under uncertainty, be innovative, be imaginative and creative.”

- “*Human have consciousness, the fact of awareness by the mind of itself and the world.*”
- “*Super or strong AI also known as Artificial Super Intelligence (ASI)*” is applied to a level of Intelligence of systems/machines that could surpass the human intelligence, and they could perform any task better than a human with cognitive properties. Some keys for this hypothetical concept of “*Strong AI*” include the ability to think, reason, make judgments, plan, learn, and communicate on their own. The concept of the “*ASI system*” is based on:
 - “*An imaginary supersystem/computer that will surpass human intelligence in all aspects,*” from creativity, to general wisdom, to problem-solving.
 - “*A hypothetical superagent that possesses intelligence far surpassing the brightest and most gifted human minds.*”

One breakthrough example of “*AI systems trying to behave as an AGI system*” was the development of the computer program: *AlphaGo* [15], although it was *designed only to play Go*. “*Go*” is an abstract strategy board game for two players invented in China more than 3000 years ago, in which the rules are simple:

- Players take turns to place black or white stones on a board of 19×19 ;
- The objective is to try to capture the opponent’s stones; or
- Surround empty space to make points of territory.

“*Go*” is a game of profound complexity, there are 10^{170} possible board configurations, making “*Go a googol* (1.0×10^{100}) *times more complex than chess.*” “*AlphaGo*” was initially trained on thousands of human amateur and professional games to learn how to play “*Go,*” but the version “*AlphaGo Zero*” learned to play the game of *Go* simply by playing games against itself. In other words “*AlphaGo Zero*” became its own teacher by applying “*Reinforcement Learning algorithms,*” starting from completely random play; it was trained without any human interference such as input data or labels. In doing so, it surpassed the performance of all previous versions, including those which beat the World Go Champions Lee Sedol and Ke Jie, becoming arguably the strongest *Go* player of all time.

In my modest opinion:

- Currently, scientists are far away from building a Strong AI, but worldwide researchers are continuously creating AI systems based on complex single or multiple Narrow AI systems. They are trying to develop General AI

(Continued)

(Continued)

systems, but, based on the resources needed that are not yet available today, this is going to take time, but how long is the big question.

- Humans have a special and nonreproducible gift from God; dimensionless from $-\infty$ to $+\infty$, abstractness, creativeness, kindness, spirituality that cannot be reached, which can be defined as “*ASI*.” Nevertheless, there is the danger with “*AI*,” that it can handle our daily activities, important decisions, and may cause more damage to us if we were to apply poor judgments when using and applying AI systems in applications that can control us and guide us, leading to the loss of the great meaning of our lives.

1.3.3 Basic types of AI systems based on functionality

Another way of recognizing the “*AI types is based on their functionality*,” as shown bottom right in Fig. 1.2. These are: *Reactive Machines*, *Limited Memory*, *Theory of Mind*, and *Self-Awareness* [16]. Where:

- “*Reactive Machines*” type systems are a simple type of “*AI*,” which does not have the ability to form memories or to use experience to take a decision. “*Reactive machines*” have an intelligence that only perceives the world directly and acts on what it sees; they cannot function beyond the specific task for which they were designed. These machines will behave exactly the same way every time they encounter the same situation [17].

One example is the “*Deep Blue IBM’s chess-playing supercomputer*,” as indicated in Fig. 1.3. *Deep Blue* beat the world grandmaster Garry Kasparov in the late 1990. “*Deep Blue*” can identify the pieces on a chess board and know each move. It can make predictions about what moves might be next for it and its opponent. And it can choose the most optimal moves from among the possibilities. But it does not have any concept of the past, nor any memory of what has happened before. Apart from a rarely used chess-specific rule against repeating the same move three times, “*Deep Blue*” ignores everything before the present moment.

- “*Limited Memory*” type AI systems can store past experiences or some data for a limited short period of time and apply them to a preprogrammed representation of their surrounding environment. For example, self-driving cars can store the recent speed of nearby cars, the distance of other cars, the speed limit, and other information to navigate the road. These observations are added to the “*self-driving cars*” preprogrammed representations of the world, which also include lane markings, traffic lights, and other important elements like curves in the road. They are included when the car decides when to change lanes to avoid cutting off another driver or being hit by a nearby car. But these simple pieces of information about the past are only transient. They are not saved as part of the car’s library of experience from which it can learn, the way human drivers compile experience over years behind the wheel.
- “*Theory of Mind*” type AI systems will be the machines based on *psychology* concepts for

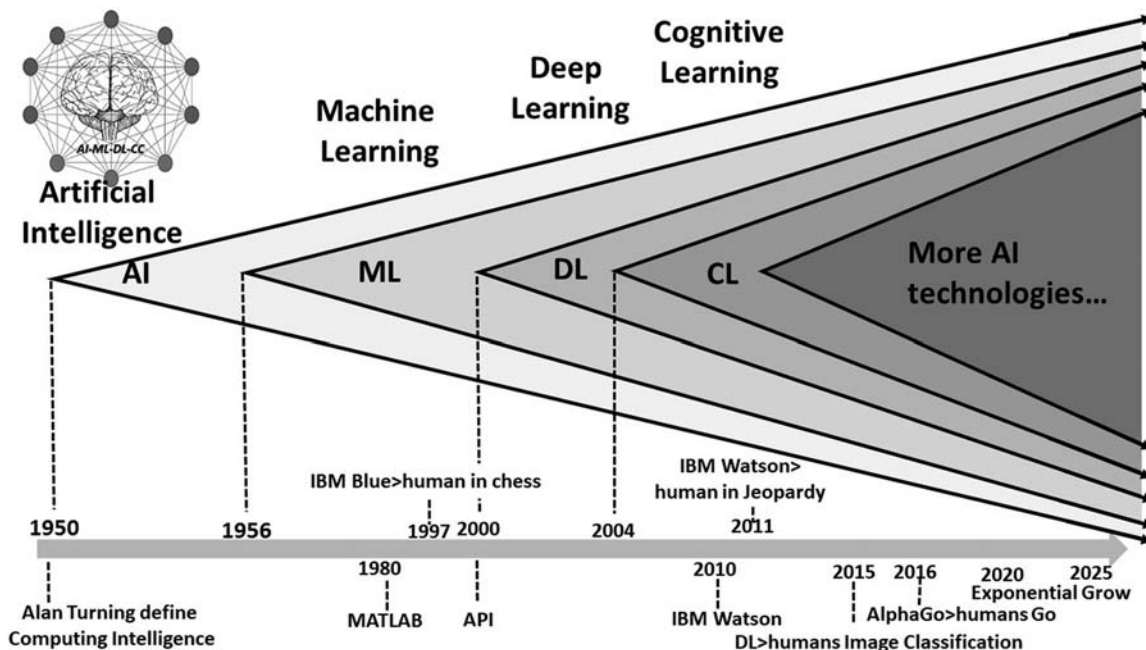


FIGURE 1.3 Artificial Intelligence evolution includes ML, ML includes DL, DL includes CC (or CL).

understanding people, creatures, and objects in the world, which can have thoughts and emotions that affect their own behavior. If this kind of AI systems is designed to walk around us, and have thoughts, feelings, and expectations for how we should be treated, they will have the capability to adjust their behavior according to each situation.

- “*Self-Awareness*” type of AI systems will be the machines that form representations of themselves and have consciousness as an extension of the “*theory of mind types*.” Conscious beings will be aware of themselves, know about their internal states, and will be able to predict the feelings of others.

Currently, *Reactive Machines* and *Limited Memory AI* types have achieved success, whereas researchers around the world are working on “*Theory of Mind and Self-Awareness machines AI types*.” However, the most successful projects are based on the integration of many AI technologies, that is, projects melding humans and AI machines and other revolutionary concepts.

1.3.4 AI technology evolution

We can detect that the “*AI technology evolution*” is on its way and growing, based on the availability of resources needed for its evolution, as indicated in Fig. 1.3. Each AI technology can be seen as a set of algorithms that give a smart system a special way of behaving; these are AI, ML, DL, CC, and others. Where each can be defined as:

- “*Artificial intelligence (AI)*” is an area of “*computer science*” that emphasizes the creation of intelligent machines that work and react like humans.

The algorithms for AI models are studied in this book in Chapter 3, Artificial Intelligence Models Applied to Biomedical Engineering. In the application of AI, algorithms are directed to resolve Biomedical Engineering problems through “*Evolutionary Algorithms*” that emulate natural evolution, such as “*Genetic Algorithms*,” “*Swarm Algorithms*,” “*traditional search methods*,” “*optimization of numeric value problems in 2D and 3D*,” and “*visual analysis of Biomedical Engineering datasets of different diseases from different Bioinstruments to analyze relation between their attributes*” by applying “*AI tools*.” There are examples and tutorials in MATLAB and IBM Watson Studio SPSS Model Flow.

- “*Machine Learning*” is a subset of “*AI*,” and it defined as the scientific study of algorithms and statistical

models that computer systems use in order to perform a specific task effectively without having to use explicit instructions, relying on patterns and inference.

The algorithms for ML models are studied in this book in Chapter 4, Machine Learning Models Applied to Biomedical Engineering. “*Machine Learning*” is studied as a subset of “*AI*” following the steps to obtain a “*prediction model*” based on “*pattern recognition*” in data using: “*clustering*,” “*classifiers*” and “*regression*” models. Some advice to apply the most appropriated “*ML Model*,” is to pre-analyze the actual problem to resolve and deduct which type of machine learning fit best for the answer needed. Generally, it is based on understanding very well the different between ML types available as “*Unsupervised Learning*,” “*Supervised Learning*,” “*Reinforcement Learning*,” “*Survival Models*,” “*Association Rules*,” and others. The study of different “*ML Models Families*” in this book are based on available ML models from “*IBM Watson SPSS Modeler Flow/ IBM Watson Machine Learning applications*” and “*MATLAB ML solution under the Statistics and Machine Learning Toolbox*” , applied in research tutorial examples to be analyzed with the purpose of obtain prediction AI models for different biomedical datasets as: “*Heart diseases*,” “*Kidney diseases*,” “*Breast cancers*,” and “*Diabetes Mellitus*.”

- “*Deep Learning (DL)*” is a subset of “*ML*,” and it is defined as the algorithms inspired by the structure and function of the “*human brain*,” such as the “*Artificial Neural Networks (ANN)*” that are designed to recognize patterns that are represented by numeric vectors that represent images, sound, text, or time series.

The algorithms for DL models are studied in this book in Chapter 5, Deep Learning Models Principles Applied to Biomedical Engineering and in chapter 6, Deep Learning Models Evolution Applied to Biomedical Engineering.

- Chapter 5, Deep Learning Models Principles Applied to Biomedical Engineering. The underlying principle of “*Deep Learning*” is a compositional nature of “*neural network*” inspired by the biological elements that forms the “*human brain*,” as a collection of “*nodes*” emulating “*brain neurons*,” and their “*neuron synapses*” connections as primary elements of a net, that combine to form mid-level elements identified as “*Artificial Neural Networks (ANN)*,” which in turn are combined with different architectures to form more complex networks. In this book “*ANN*” are organized based on their architectural type, and the way their different components are connected to one another define the specific learning goal with

(Continued)

(Continued)

different types such as “Feed Forward Neural Network,” “Backpropagation Neural Networks,” “Recurrent Neural Networks,” “Memory Augmented Neural Networks,” “Modular Neural Networks,” and “Evolutive Neural Networks.” The first two “ANN types” are studied in this chapter along with a general net “Shallow Neural Network” and a special kind of learning known as “Transfer Learning from pretrained Deep Learning Networks.” All are based on examples and practical research of Biomedical Engineering using existing “AI tools” from: “MATLAB and “IBM Watson Studio.”

- **Chapter 6**, Deep Learning Models Evolution Applied to Biomedical Engineering. This chapter focuses on the study of “Deep Learning Models Evolution” which combines mid-level elements with different connections of “ANN” to form more complex networks types, such as “Recurrent Neural Networks,” “Memory Augmented Neural Networks,” “Modular Neural Networks,” and “Evolutive Neural Networks.” Many research examples are explained in this chapter applied to Biomedical Engineering solutions.

- **Cognitive Computing (CC) or Cognitive Learning (CL)** is special applications that are implemented using any combination of “AI” technologies to build “cognitive models” that “mimic human thought” processes using “NLP,” “handwriting recognition,” “face identification,” “behavioral pattern determination” and other special algorithms, such as “sentiment analysis.” “CC” is used to assist humans in their decision-making process and in this book are proposed as methods to evaluate “human cognitive status.”

Note: In the rest of this book, the term “Cognitive Computing abbreviated CC” is synonymous with “Cognitive Learning abbreviated CL”.*

The algorithms for CC models are studied in this book in **Chapter 2**, Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to Help in the Solution of AI Biomedical Engineering Problems, and **Chapter 7**, Cognitive Learning and Reasoning Models Applied to Biomedical Engineering.

- **Chapter 2**, Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to Help in the Solution of AI Biomedical Engineering Problems. This chapter introduces the analysis for “nonmotor symptoms (related to cognition and no related to movement disorders); “Human cognitive development stages” and their relation to “brain neurons and neural pathways”; “Cognition and its
(Continued)

(Continued)

integration with multidiscipline sciences.” “Natural Language Processing applications,” examples and/or exercise for NLP Topics, Audio Labeler for Machine Learning, NLP Text to speech, NLP Speech to Text, NLP analysis for: Sentiment, Emotion, Keywords, Entities, Categories, Concept and Semantic Roles with MATLAB and API as a set of functions and procedures allowing the creation of applications that access the features or data of an operating system, application, or other service through IBM Cloud services.

- **Chapter 7**, Cognitive Learning and Reasoning Models Applied to Biomedical Engineering. In this chapter we focus on many “preudies and preanalysis of different Biomedical Engineering problems that need to be develop with specialized research projects applying Cognitive Learning and Reasoning (CL&R) algorithm, that can be integrated to the Proposed General Architecture framework of a Cognitive Computing Agents System (AI-CCAS)” with special emphasis on “Cognitive Learning and its relationship with neuroscience of reasoning proposed as CL&R using CC.” In this book, we have studied many interactions of different human illness, diseases, and disorders, where “human illness” is defined as body damage that needs to be cured, such as infections, injuries, cells degeneration, etc.; “human diseases” are defined as states or reactions that must be managed, such as pain, discomfort, weakness, fatigue, etc.; and “human disorders” are defined as functions or abnormalities that must be treated, such as physical, mental, genetic, emotional/behavioral, and functional disorders. The complexity for the analysis needed is boundless and can only be analyzed through the multidisciplinary sciences, where interactions of science such as “biomedical engineering,” “neurology,” “cognitive sciences” and “computer science” using tools with “exponential technologies” such as AI and others through “continuous exponential evolution” that includes ML, DL, and CC. The main purpose is obtaining useful “AI models” that can help analyze to human health problems. Now it is the time to apply them and many others in research projects.

In summary, we can deduce that in general terms: “CC is a subset of DL,” and “DL is a subset of ML,” and “ML is a subset of AI,” as shown in Eq. (1.4).

$$AI \text{ subsets} : CC \subseteq DL \subseteq ML \subseteq AI \quad (1.4)$$

where \subseteq means a subset

If Artificial Intelligence is any system/machine/computer program that can imitate intelligent human behavior, then all CC applications are AI, but not all AI are CC.

1.3.5 AI in industries

“AI” is revolutionizing almost all industries to enhance their business like never before with thousands of “AI services” and “AI products.” Some are manufacturing, supply chain, human resources, customer services, marketing, advertising, building management, automotive, agriculture, and medicine:

- “*Manufacturing processes*” always can be improved by applying “*AI technologies*”; even when machines do some of the labor, by analyzing continuously many issues, such as predictive and preventive maintenance, enhancing testing machines and robot manufacturing effectiveness, quality management, mass customization, and many more processes. Currently, “*Industry 4.0**” is a name given to the idea of smart factories where machines are augmented with web connectivity and connected to a system that can visualize the entire production chain and make decisions on their own. The trend is toward automation and data exchange in manufacturing technologies which currently include: “*Cyber-Physical Systems (CPS)*,” “*Internet of things (IoT)*,” “*Industrial Internet of Things (IIOT)*,” “*Cloud Computing*” [18], and others,

Note:* Industry 4.0 is also referred to as the fourth industrial revolution [19].

- “*Supply Chain processes*” have numerous uncontrollable factors, such as weather, delivery delays, unstable suppliers, and others that can create issues for an entire company. Due to large amounts of information needing to be processed at any given time, AI technologies can analyze, detect, and give faster and better solutions. Currently, some “*AI supply*” chain solutions are already revolutionizing, such as “*IBM Watson Supply*,” “*Watson supply chain Insight*,” and others.
- “*Human Resources Processes*” can be improved using “*AI technologies*,” such as reviewing hundreds of resumes a day, analyzing and creating scores for each candidate to facilitate the hiring decision based on up-to-date statistical facts, and removing the possibility of bias. Currently, there are *AI solutions*, such as “*IBM Watson Recruitment*,” “*IBM Watson Career Coach*,” and others.
- “*Customer Services processes*” can be improved by gathering and analyzing a company’s data from various sources and the previous cases and issues, with the objective of enriching and efficient interactions with customers. Currently, “*AI solutions*” for this industry are “*Watson Discovery for Salesforce*,” “*Zendesk*,” and others.
- “*Marketing processes*” can be improved by understanding and predicting how a customer behaves when

shopping and give smart recommendations for target audiences. Currently, “*AI solutions*” for this industry are “*IBM Marketing Solutions*,” “*IBM Watson Marketing Insights*,” and others

- “*Advertising process*” can be improved by utilizing dynamic creative tools that are aware of the weather, time of day, location, consumer behavior, etc., to deliver personalized ads to customers. Currently, “*AI solutions*” are “*Google advertising*,” “*IBM Watson Advertising*,” and others.
- “*Building Management*” can be improved by analyzing how the buildings are operating with sensors that give the information in real time, reducing maintenance costs, improving safety, increasing sustainability, and optimizing their functionality. Currently, “*AI solutions*” are “*IBM IoT Building Insights*,” and others.
- “*Automotive industry*” can be improved with “*IoT sensors*” that report vehicle issues in real time, analyzing and taking actions as quickly as possible for a more efficient behavior of the transportation vehicle. Currently “*AI solutions*” for this industry are “*IBM Watson IoT*,” “*IBM Watson Assistant for Automotive*,” and others.
- “*Agriculture processes*” can be improved by creating an electronic field record with up-to-date data, such as weather with satellite image or drones, soil reading, crop health analysis to take actions for crop protection, higher-quality crops to help eradicate extreme poverty and hunger in the world. Currently “*AI solutions*” for this industry are “*Watson Decision Platform for Agriculture*” and others.
- “*Medicine and healthcare*” also always need to improve processes by applying “*AI Technologies*” for resources for hospitals, clinics, doctors, researches, patients, consumers, etc. Some examples are:
 - *leverage data*: analyzing data and making cross-references with previous cases and studies;
 - *optimizing collaboration*: sequencing of the human genome with partnerships with diagnostics labs and research centers;
 - *human resources*: optimizing care programs by increasing coordination, detecting inefficient processes, defining priorities on value-based healthcare; and
 - many more processes.

This book focuses on AI applied in Biomedical Engineering and the fields/subfields involved, such as medicine, biology, healthcare services, and related fields, with the purpose of obtaining models for analysis, classification, forecasts; to be used to confirm diagnostics and therapeutics, making special emphasis in neurologic diseases with nonmotor symptoms.

1.4 Machine learning

As shown in Fig. 1.2, and stated in Eq. (1.4), “*ML*” is a subset of “*AI*” that provides systems with the ability to automatically learn and improve from experience without being explicitly programmed. We can differentiate “*AI*” and “*ML*”:

- “*AI*” is a broader concept of machines that are being to carry out smart tasks,
- “*ML*” is a subset of “*AI*” that gives machines access to data and let them learn from themselves.

“*ML*” are machines that learn for themselves from the data history. “*ML*” has as a primary objective the discovery of patterns in the data, to create a model that is useful for taking decisions from the data. “*ML algorithms*” build a mathematical model based on “*training data*” from sample data, to make predictions or decisions without being explicitly programmed to perform the task [20]. “*ML*” is a branch of “*computer science*” and applies multidisciplinary fields and different processes to reach its objectives, these are: “*statistics*,” “*computational statistics*,” “*databases*,” “*know discovery database*,” “*data mining*,” “*mathematical optimization*,” “*exploratory data analysis*,” and others, where:

- “*Statistics*” is the science of collecting and analyzing numerical data in large quantities, especially for the purpose of inferring proportions in a whole from those in a representative sample.
- “*Computational statistics (also known as statistical computing)*” is the area of computational science specifically dedicated to the study of the mathematical science of statistics. It is the interface between statistics and computer science.
- “*Database*” is an organized collection of data. It is the collection of schemas, tables, queries, reports, views, and other objects. The data are typically organized to model aspects of reality in a way that supports processes requiring find information.

- “*Data mining*” has the goal of extracting information using intelligent methods from a large dataset to discover patterns and useful knowledge.
- “*Know discovery database (KDD)*” is the process of discovering useful knowledge from databases. It is a data mining technique that includes data preparation and selection, data cleansing, incorporating prior knowledge on datasets and interpreting accurate solutions from the observed results.
- “*Mathematical optimization or mathematical programming*” is the selection of a best element from some set of available alternatives.
- “*Exploratory data analysis (EDA)*” is a statistics approach to analyzing datasets to summarize their main characteristics, often with visual methods. A statistical model can be used or not, but primarily “*EDA*” is for seeing what the data can tell us beyond the formal modeling or hypothesis testing task.

“It is very important to point out that this book makes frequently uses of these multidiscipline fields to reach the objectives of “Artificial Intelligence”; the magic of “AI” is based specially on “mathematics” that is used to define each “AI model,” “statistics” that help in the criteria for validate if a model is accepted or not, and “data mining” to extract important information from the databases.”

1.4.1 ML seven specific steps

“*ML*” has seven general steps to achieve its goal of obtaining a valid model for prediction. These steps are shown in Fig. 1.4 and they are: “*data collection*,” “*data preparation and exploration*,” “*feature engineering*,” “*model selection*,” “*model training*,” “*model evaluation*,” and “*model prediction and deployment*.”

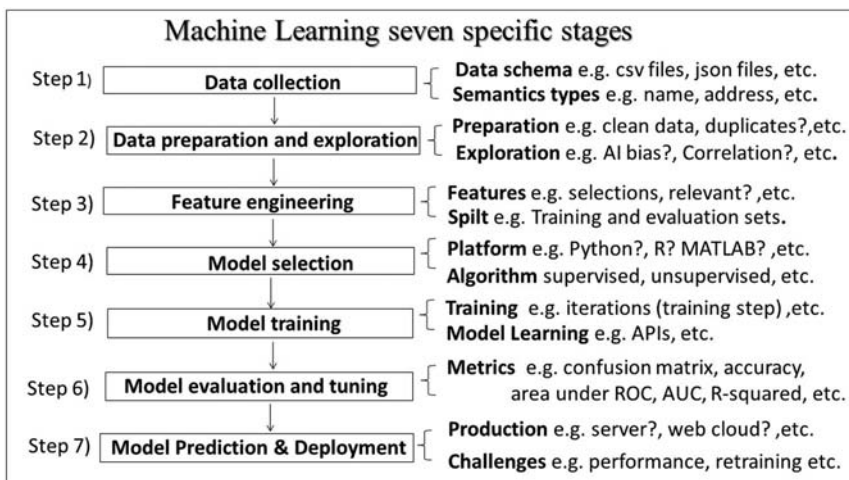


FIGURE 1.4 Machine Learning—seven general steps for an AI model to achieve predictions.

1.4.1.1 Step 1) Data collection

“Data collection” is very important because the quantity and quality of the data dictate how accurate the model will be. “Data collection” is made under two terms: *data schema* and *semantic types*, where:

- “Data schema” is a structure for organizing the data. It defines both the data contents and relationships. The most common data schemas formats are “csv” as comma-separated values are tabular data such as spreadsheet or database, and “json” as JavaScript Object Notation is a lightweight data-interchange format, it is used primarily to transmit data between a server and web application, as an alternative to XML.
- “Semantic types” are the labels that are specified for each column of the data in tabular form, that is, Address, Name, Phone, Income, etc.

Note*: “Precollected data” can be used too for this step of data collection, these are “web sources datasets,” such as “Kaggle,” an open datasets web platform for data science at <https://www.kaggle.com>, “UCI Machine Learning Repository” in <https://archive.ics.uci.edu/ml/datasets.php>, and others.

1.4.1.2 Step 2) Data preparation and exploration

- “Data preparation” is the cleaning of data collected by applying the following substeps: clean-tools to remove duplicates, correct errors, deal with missing values, normalization, data type conversions, etc., and randomize data, which erases the effects of the particular order in which the data are collected and/or otherwise prepared.
- “Data exploration,” is a very important tool for avoiding “AI bias”; where, “AI bias” is a phenomenon that occurs when an algorithm produces results that are systematically prejudiced due to erroneous assumptions in the ML process. Some of the tools are:
 - statistical tools for the measuring of tendency, dispersion, and shape, such as mean, median, standard deviation, variance, skewness, kurtosis, etc.;
 - correlation to help detect relevant relationships between variables or class imbalances, between variables; and
 - visualizing of the data distribution plots as histograms, boxplots, bar plots, etc.

1.4.1.3 Step 3) Feature engineering

“Feature engineering consists of:

- Feature selection includes the deletion of noisy or irrelevant attributes;
- Determining which features are more relevant and even creating new features;

- Applying dimensionality reduction using special tools, such as Principal Component Analysis (PCA);
- Finally, splitting the dataset into *training* and *evaluation sets*. The *training set* is used to build a model, while the *evaluation set* is used to validate the model built. “It is important to not include the same data points of the training set in the test (evaluation) set.”

1.4.1.4 Step 4) Model selection

Model selection refers to the selection of the best algorithm and the platform, which could be “open source language,” “high-performance language,” or “special AI cloud applications,” where:

- “Open source language,” such as “Python language,” a free AI and general-purpose programming language, or “R language” a free AI data-statistical analysis programming language.
- “High-performance language,” such as MATLAB, Lisp, Prolog, and others.
- “Special AI cloud applications,” such as the IBM Cloud Watson computer and others.

The algorithm to select depends on the type of machine learning problem, generally this can be “Supervised Learning,” “Unsupervised Learning,” “Reinforcement Learning,” “Survival Models,” and “Association Rules,” where each type is as follows:

- “Unsupervised Learning” is used when the data does not include the result for each case, meaning that that the ground truths are unknown. These unsupervised algorithms can find patterns in a stream of inputs. Some commons algorithms are based on the “clustering technique,” such as “k-means,” “k-modes,” “k-prototypes,” “DBScan,” “Expectation Maximization,” and others.
 - “k-means clustering” is an “ML-Unsupervised Learning algorithm” that uses “vector quantization” and obtains “k clusters” in which each observation belongs to the cluster with the nearest mean.
 - “k-modes clustering” is an “ML-Unsupervised Learning algorithm” that uses “modes”: where “modes” means having highest frequency, instead of means to form clusters of categorical data. In other words, “k-modes algorithm” distance is measured by the number of common categorical attributes shared by the two data points.
 - “k-prototypes” is an “ML-Unsupervised Learning algorithm,” which is the simple combination of “k-means and k-modes” in clustering mixed attributes of numerical and categorical values.
 - “DBScan” is an “ML-Unsupervised Learning algorithm” for “Density-Based Spatial clustering” of applications with noise. “DBScan” is one of the

algorithms in which a density-based clustering method is used to detect outliers.

- “*Expectation maximization*” is an “*ML-Unsupervised Learning algorithm*” that uses an iterative method to “*find maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models,*” where the model depends on unobserved latent variables.
- “*Recommendation systems*” is an “*ML-Unsupervised Learning algorithm*” that is a subclass of an information filtering system that seeks to predict the “*rating*” or “*preference*” that a user would give to an item, applying matrix factorization and collaborative filtering.
- “*Supervised Learning*” is used when data include a result for each case. This kind of algorithm is used for “*classification and regression,*” where:
 - “*Classification*” is a method used to determine what category or class something belongs to, after seeing several examples of things from several categories; the final category is a discrete variable.
 - “*Regression*” is a method that attempts to produce a function that describes the relationship between inputs and outputs and predicts how the outputs should change as the inputs change; the prediction is a real number.

Some of the most common supervised learning algorithms are “*decision tree,*” “*random forest,*” “*logistic regression,*” “*regression,*” “*support vector machines,*” “*neural nets,*” and others, where:

- “*Decision tree*” is an “*ML-Supervised Learning algorithm*” decision support that uses a tree-like graph or model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.
- “*Random forest*” is an “*ML-Supervised Learning algorithm*” that constructs a multitude of “*decision trees*” at training time and outputting the class that is the mode of the classes applying classification or mean prediction applying regression of the individual trees.
- “*Logistic regression*” is an “*ML-Supervised Learning algorithm*” that uses the logistic function to predict a binary class.
- “*Regression*” is a set of “*ML-Supervised Learning algorithms,*” such as “*Linear regression,*” “*GLM regression (Generalized Linear Models),*” “*Least Squares regression,*” etc.
- “*Support Vector Machines (SVM)*” is an “*ML-Supervised Learning algorithm*” that performs classification by finding the hyperplane that maximizes the distance margin between the two classes. The extreme points in the datasets that define the hyperplane are the support vectors.

- “*Artificial Neural Networks*” is an “*ML-Supervised Learning algorithm*” with multilayers perceptron as the “*backpropagation algorithm*” and others such as “*Deep Learning algorithms*” based on bigger and complex neural networks.
- “*Reinforcement Learning*” is an “*ML algorithm*” inspired by behaviorist psychology, concerned with how software agents ought to take actions in an environment to maximize some notion of “*cumulative reward.*”
- “*Survival Models*” is an “*ML algorithm*” that is used to analyze data in which the time until the event is of interest. The response is often referred to as a failure time, survival time, or event time.
- “*Association Rules*” is an “*ML algorithm*” that is a rule-based machine learning method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using some measures of interestingness.

1.4.1.5 Step 5) Model training

“*Model Training*” is running the algorithm to obtain a process model based on iteration as a training step using the actual dataset, to train the model for performing various actions. This is the actual data in the ongoing development process models learning with various “*Application Programming Interfaces* (API)*” and algorithms to train the machine to work automatically.

Note:* API in general terms is a set of clearly defined methods of communication among various components. Examples of APIs will be studied in Chapter 2, *Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to Help in the Solution of AI Biomedical Engineering Problems, of this book.*

1.4.1.6 Step 6) Model evaluation and tuning

“*Model evaluation*” is the use of metrics or a combination of them to measure the objective performance of the model and to select the best algorithm for that specific purpose. The most common techniques are “*cross-validation,*” “*confusion matrix,*” “*precision and recall,*” “*ROC,*” “*AUC,*” “*Root Squared error,*” and “*Accuracy,*” where:

- “*Cross-validation*” is a resampling method used for model evaluation to avoid testing a model on the same dataset on which it was trained.
- “*Confusion matrix*” is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning, it is usually called a “*matching matrix.*”
- “*Precision and recall,*” where:
 - I. “*Precision*” is the proportion of positive identifications that was actually correct. A model that

produces no false positives has a precision of 1. “Precision” is calculated by applying Eq. (1.5).

$$\text{Precision in ML } \text{Precision} = \frac{TP}{TP + FP} \quad (1.5)$$

where TP are the True Positives, FP are the False Positives.

- II. “Recall” is the proportion of actual positives that were identified correctly. A model that produces no false negatives has a recall of 1. “Recall” is calculated by applying Eq. (1.6).

$$\text{Recall in ML } \text{Recall} = \frac{TP}{TP + FN} \quad (1.6)$$

where TP are the True Positives, FN are the False Negatives.

- “ROC (Receiver Operating Characteristic),” where the “ROC curve” is a graphical plot that summarizes how a classification system performs and allows us to compare the performance of different classifiers. The “ROC curve” plots two parameters: “True Positive Rate (TPR)” and “False Positive Rate (FPR).”

- I. “TPR” values are shown in the vertical axis of the “ROC curve” and are calculated using the Eq. (1.7). Another useful parameter is “specificity rate” as shown in Eq. (1.8).

$$\text{True Positive Rate } TPR = \text{Recall} = \frac{TP}{TP + FN} \quad (1.7)$$

where TP are the True Positives, FN are the False Negatives.

$$\text{Specificity rate} = \frac{TN}{TN + FP} \quad (1.8)$$

where TN are the True Negative, FP are the False Positives.

- II. “FPR” values are shown in the horizontal axis of the “ROC curve” and are calculated using Eq. (1.9).

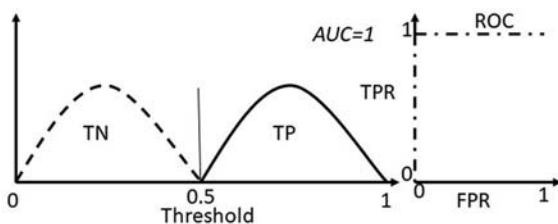
$$\begin{aligned} \text{False Positive Rate } FPR &= 1 - \text{Specificity} \\ &= \frac{FP}{FP + TN} \end{aligned} \quad (1.9)$$

where FP are the False Positives, TN are the True Negatives.

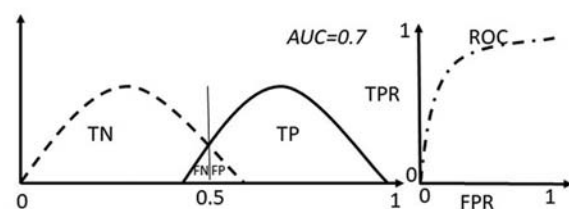
- “Area Under the Curve (AUC)” is a measurement of the area under the entire function “ROC curve” in all the range, where the “AUC” values range from 0 to 1. “AUC” represents the probability that the evaluated AI model ranks a positive example more highly than a random negative example. In other words, “AUC = 0” means that the model’s predictions are 100% wrong, and “AUC = 1” means that the model’s predictions are 100% correct.

“It is important to differentiate between AUC and ROC curve: ROC is a probability curve and AUC represents the degree of separability.” To understand these two concepts four cases of AUC and ROC and their relationships are shown in Fig. 1.5.

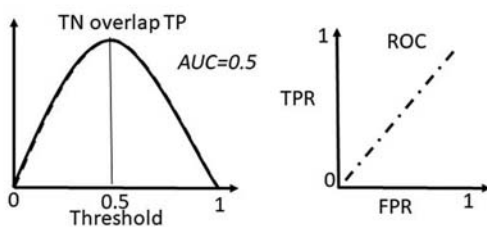
I. Ideal Case: AUC=1 with threshold=0.5



II. Acceptable case: AUC=0.7 with threshold=0.5



III. Worst case: AUC=Threshold=0.5



IV. Unacceptable case: AUC=0 and Threshold=0.5

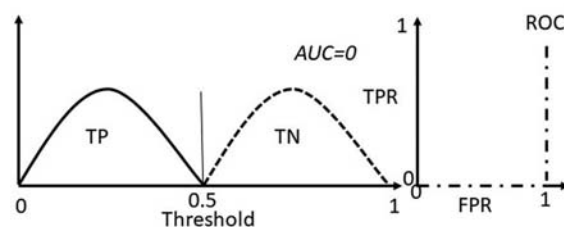


FIGURE 1.5 Analysis of a model using AUC and ROC: (I) ideal case, (II) acceptable case, (III) worst case, and (IV) unacceptable case.

- I. “*Ideal case: AUC = 1 with threshold = 0.5,*” this is when the two probability distribution curves do not overlap and the model has a perfect way of distinguishing between *TN* and *TP*.
- II. “*Acceptable case: AUC = 0.7 with threshold = 0.5,*” this is when the two probabilities’ distributions overlap a reasonable value, and $AUC > Threshold$. This means that there is 70% chance that the model distinguishes between *TN* and *TP*.
- III. “*Worst case: AUC = threshold = 0.5,*” this is where both probabilities’ distribution curves overlap 100%, this is an indication that the model does not have a way to differentiate between *TN* and *TP*.
- IV. “*Unacceptable case: AUC = 0 and Threshold = 0.5,*” in this case the model has the same chance of predicting a *TN* as a *TP* or vice versa.
 - “*R-squared error (R^2),*” also known as the “*coefficient of determination,*” is a statistical measure that represents the proportion of the variance for a dependent variable that is explained by an independent variable or variables in a regression model. Whereas *correlation* explains the strength of the relationship between an independent and dependent variable, “*R-squared error*” explains to what extent the variance of one variable explains the variance of the second variable. So, if the R^2 of a model is 0.50, then approximately half of the observed variation can be explained by the model’s inputs, based on the range: $0 \leq R^2 \leq 1$. R^2 is calculated using Eq. (1.10).

$$R^2 = 1 - \frac{\text{Explained Variance}}{\text{Total Variance}} \quad (1.10)$$

$$= 1 - \frac{SS_{res}}{SS_{tot}}$$

Explained Variance (SS_{res}) and *Total Variance* (SS_{tot}) are calculated by the following steps:

- “*Total Variance,*” also known in statistics as “*total sum of squares* (SS_{tot}),” is calculated by subtracting the average actual value from the predicted values, squaring the results and summing them. These steps are stated in Eq. (1.11).

Total Variance or total sum of square

$$SS_{tot} = \sum_i (y_i - \bar{y})^2 \quad (1.11)$$

- “*Explained Variance,*” also known in statistics as “*explained sum of squares* (SS_{reg}),” is calculated by taking the data points (observations) of dependent and independent variables and finding the line of best fit, often from a regression model. From there you would calculate

predicted values, subtract actual values and square the results. This yields a list of errors squared, which is then summed and equals the explained variance. These steps are stated in Eqs. (1.12) and (1.13).

Explained sum of squares

$$SS_{reg} = \sum_i (f_i - \bar{y})^2 \quad (1.12)$$

where f_i are the predicted values and $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$ is the mean of the n observed data.

Residual sum of squares

$$SS_{res} = \sum_i (y_i - f_i)^2 = \sum_i e_i^2 \quad (1.13)$$

where y_i are the values of the dataset and f_i are the predicted values.

In some cases, the “*total sum of squares*” equals the sum of the two other sums of squares defined above, as shown in Eq. (1.14).

$$\text{Total sum of squares } SS_{tot} = SS_{res} + SS_{reg} \quad (1.14)$$

- “*Accuracy*” is one metric for evaluating classification models, analyzing the fraction of predictions the ML model got right using the Eq. (1.15).

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \quad (1.15)$$

$$= \frac{TP + TN}{TP + TN + FP + FN}$$

where TP = True Positives, TN = True Negatives, FP = False Positives, and FN = False Negatives.

Once the evaluation of the model is finished, an improvement can be made by “*tuning the model*” also known as “*hyperparameter optimization.*” A “*hyperparameter*” is a parameter whose value is used to control the learning process; this is different to other parameters, such as the typically node weight, that are learned. The hyperparameters give a way for model optimization that minimizes a predefined “*loss function,*” also known as cost on given independent data [21]. The “*hyperparameters*” are often set by heuristics or tuned for a given predictive modeling problem. For example, when a “*machine learning algorithm*” is tuned for a specific problem, such as when you are using a “*grid search*” or a “*random search,*” then we are tuning the “*hyperparameters of the model*” or order to discover the parameters of the model that result in the most skillful predictions. A good rule of thumb to overcome this confusion is as follows:

“If you have to specify a model parameter manually then it is probably a model hyperparameter.”

1.4.1.7 Step 7) Model prediction and deployment

“ML” is basically using data to answer questions. In this final step we get the answer for our questions; the “ML model” is used to predict the outcome of what we need. Nevertheless, building a “ML model” is not the end of a project; we need to apply the concept of “deployment,” that refers to creating applications of a model for predictions using new data where it is needed. “Deployment” means putting the “ML model in production in a computer, company server or in a web cloud,” where the model can make predictions in real time. The “deployment model” has many challenges, such as “scalability,” “performance,” “response time,” and “model retraining.”

- “Scalability in machine learning” is the models which can deal with any amount of data, without consuming tremendous amounts of resources like memory.
- “Performance value in ML” indicates how successful the predictions of a dataset have been by the training model. Usually the performance can be measured by metrics, such as “ R^2 ,” “Average Error,” “Mean Square Error,” and others. Performance must be better if we add more data, add more features, do a right feature selection, use regularization, etc.
- “Response time” is the time to obtain the prediction from the model in any specific application, usually fastest is better.
- “Model retraining” is part of the model life cycle to update the model the current data. In “Model retraining,” it is needed to run all the process again including integration of “new data,” “preprocessing,” “training,” “evaluation,” and “deployment.” It is necessary for there to be a way of automating all this process to maintain the model and to be an “ML application” that is successful with the evolution of new data.

In summary, “ML” has as a primary objective the discovery of patterns in the data, to create a model that is useful for taking decisions from new data. Every one of the seven basic steps are especially important to achieve a good prediction of the data using an “ML model.” These seven general steps must be very well understood and followed.

Note: This is only an introduction ML; this subset of AI is explained with more detail, specific examples and exercises on MATLAB and IBM Watson Studio in Chapter 4, “Machine Learning Models Applied to Biomedical Engineering,” of this book.

1.5 Deep learning

DL includes aspects of “ML algorithms,” “ANN,” and “AI.” This is explained graphically in Fig. 1.2 and stated in Eq. (1.4); where “DL” is a subset of “ML,” in which an

“AI model” learns to perform a classifications task directly from images, text, and sound. DL is usually implemented using an “ANN architecture.” The “ANN” created from these components are in the field of “AI” that comes closest to modeling the workings of the human brain. In “DL” improved mathematical formulas and increased computer processing power are enabling the development of more sophisticated applications than ever before. “DL” is also called “structured learning” and “hierarchical learning”; it is the kind of machine intelligence used to create “AI systems.” Examples are:

- *Transportation:* self-driving in vehicles, buses, taxis, airplanes, etc.
- *Voice applications,* such as search and voice-activated assistant, voice generation, music composition, etc.
- *Business applications,* such as advertising, finance, marketing, etc.
- *Robotics,* where smart robots can be trained to carry out complex tasks that require more thought and adaptation, others can learn just by observing the human task, etc.
- *Computer games,* such as GO, chess, and others.
- *Images applications,* such as image recognition, automatic image caption generation, automatic image colorization, etc.
- *Text applications,* such as automatic machine translation of text, automatic text generation, automatic handwriting generation, etc.
- *Healthcare applications,* such as smart algorithms that measure diagnose condition, advise plan treatment, and if approved by physicians it executes the delivery of treatment for different illnesses and diseases, applying neural networks for brain cancer detection, tumors, etc.
- Many more new DL applications are being created in different areas currently.

This book will focus on “DL in Biomedical Engineering applying to medicine and healthcare, that is, analyzing images, text, detecting and diagnosing medical conditions, with a special emphasis on neurologic diseases through cognitive functions.”

1.5.1 Difference between deep learning and machine learning

“DL” is a subset of “ML,” and “ML is a subset of AI.” Usually, the term “deep learning refers to deep artificial neural networks,” and somewhat less frequently to “deep reinforcement learning.” The main differences between “DL” and “ML” are:

- Traditional Machine Learning Artificial Neural Networks contain only two or three layers, while the

deep learning network can have hundreds. This is the reason for the term Deep, that refers to the number of layers in the network—the more layers, the deeper the network.

- Multiple hidden layers allow deep neural networks to learn features of the data in the so-called feature hierarchy, because simple features (e.g., two pixels) recombine from one layer to the next, to form more complex features (e.g., a line). Nets with many layers pass input data of features through more mathematical operations than nets with few layers and are therefore more computationally intensive to train.
- “ML” works fine with small and medium size datasets, their models are medium size and the computer power needed is not so intense; whereas “DL needs big data, big models, and big computational power.” “Computational intensity” is one of the hallmarks of “DL,” and it is one reason why a new kind of chip called GPUs* are in demand for the training of deep-learning models.

Note:* GPU (Graphics Processing Units) are programmable logic processors specialized for fast numeric calculation and graphic rendering, that is, NVIDIA.

Please pay special attention to differentiate and recognize algorithms and application of Deep Learning with respect to Machine Learning, cognitive computing, and Artificial Intelligence.

1.5.2 Types of artificial neural networks

“ANN” is an algorithm based on how the “human brain and the human nervous systems works.” It is based on a large collection of simple neural units known as “artificial neurons,” “loosely analogous to the observed behavior of a biological brain’s axons of the human brain*.”

Note:* For more detailed information on the human brain and neurons, please refer to Chapter 2, “Introduction to Human Neuromusculoskeletal System,” of my book *Applied Biomechanics Using Mathematical Models* [22].

Each neural unit relates to many others, and links can enhance or inhibit the activation state of adjoining neural units. Each individual neural unit computes using summation and activation function, as shown in the upper region of Fig. 1.6A. One brain neuron is connected from axon terminals to dendrites of the next neuron in a process known as “type I synapse.” Similarly, an “ANN is a massive parallel distributed processor” that has a natural propensity for storing experiential knowledge and making it available for use, as indicated in the lower region of Fig. 1.6B. It resembles the brain in two respects [23]: “Learning process” and “Interconnection strengths,” where:

- “Learning process” is the way that the knowledge is acquired by an “ANN.” “Learning” is a process in which the parameters such as the “synaptic weight” indicated, such as w_n in Fig. 1.6B, and “bias levels” of

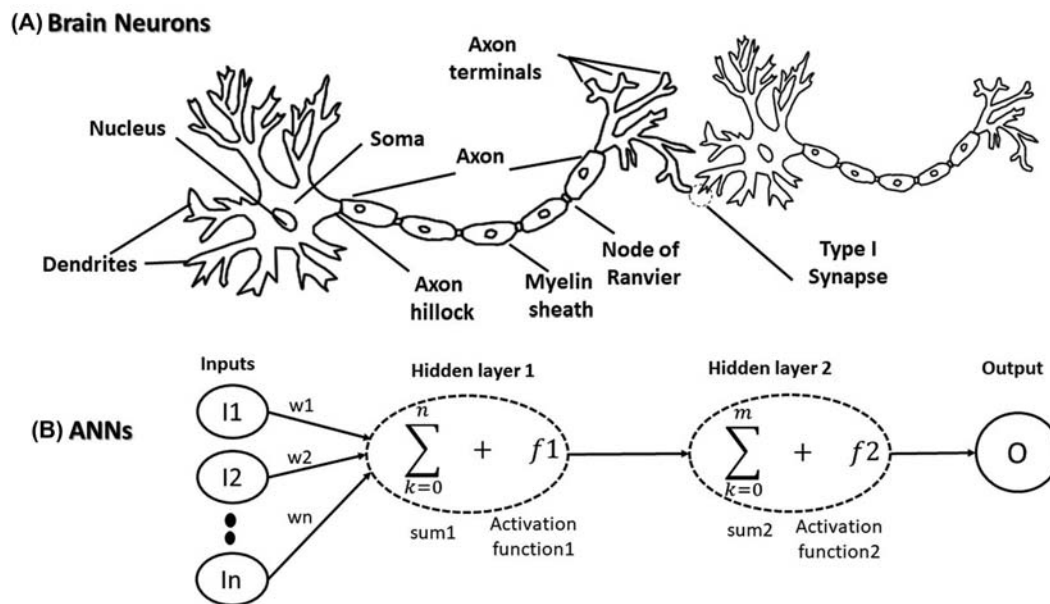


FIGURE 1.6 Artificial Neural Network (ANN) is an algorithm based on how the human brain and the human nervous systems works. Analogy: (A) two brain neurons are joined through the type I synapse versus (B) two ANNs.

a neural network are adapted through a continuous process of stimulation by the environment in which the network is embedded. The basic learning processes can be basically of two types: “Supervised learning” and “unsupervised learning.”

- “Interconnection strengths” in an “ANN” are similar to the “synaptic human weight in the brain, which are used to store the knowledge.” There may be a “threshold function or limiting function” on each connection and on the unit, itself, such that the signal must surpass the limit before propagating to other neurons.

The main advantage is that the “ANN” can be “self-learning” as an “unsupervised network,” or trained as a “supervised network,” rather than explicitly programmed, and used in areas where the solution or feature detection is difficult to express in a traditional computer program. The main disadvantage of “learning algorithms as ANN is that they may require an exponential number of iterations with respect to the number of weights until a solution to a learning task is found.” This means more processing time to obtain the desired solution [24].

There is not standard definition to classify the different types of available “ANN” based on different approaches, such as architectures and other characteristics. For the purpose of “ANNs” study, they are separated in this book based on the “methodology to process the information on the neurons (nodes) to achieve their decisions in the outputs”. These are of six different types, as indicated in Fig. 1.7:

“Feed Forward Neural Network,” “Backpropagation Neural Networks,” “Recurrent Neural Networks,” “Memory Augmented Neural Networks,” “Modular Neural Networks,” and “Evolutionary Neural Networks”.

1.5.3 Feed forward neural network

“Feed Forward Neural Network” implies a signal that can only be fed forward, meaning the absence of recurrent or feedback connections. Where the data path is only forward facing, no backward feed connections between neurons are present. Some frequently used examples of “Feed Forward Neural Network” are shown in Fig. 1.8, and these are:

“Perceptron (P),” “MultiLayer Perceptron’s (MLP) or Feed Forward Neural Network (FFN) or Deep Feed Forward Network (DFF),” “Radial Basis Network (RBF),” “Probabilistic neural network (PNN),” Extreme Learning Machine (ELM), and others.

- “Perceptron (P)” is usually a single layer neural network; it is used to classify the data into two parts. Therefore it is also known as a “Linear Binary Classifier.” It is used in “supervised learning” and helps to “classify” the given input data.

The “Perceptron (P)” is studied in more detail in Section 5.2.1.

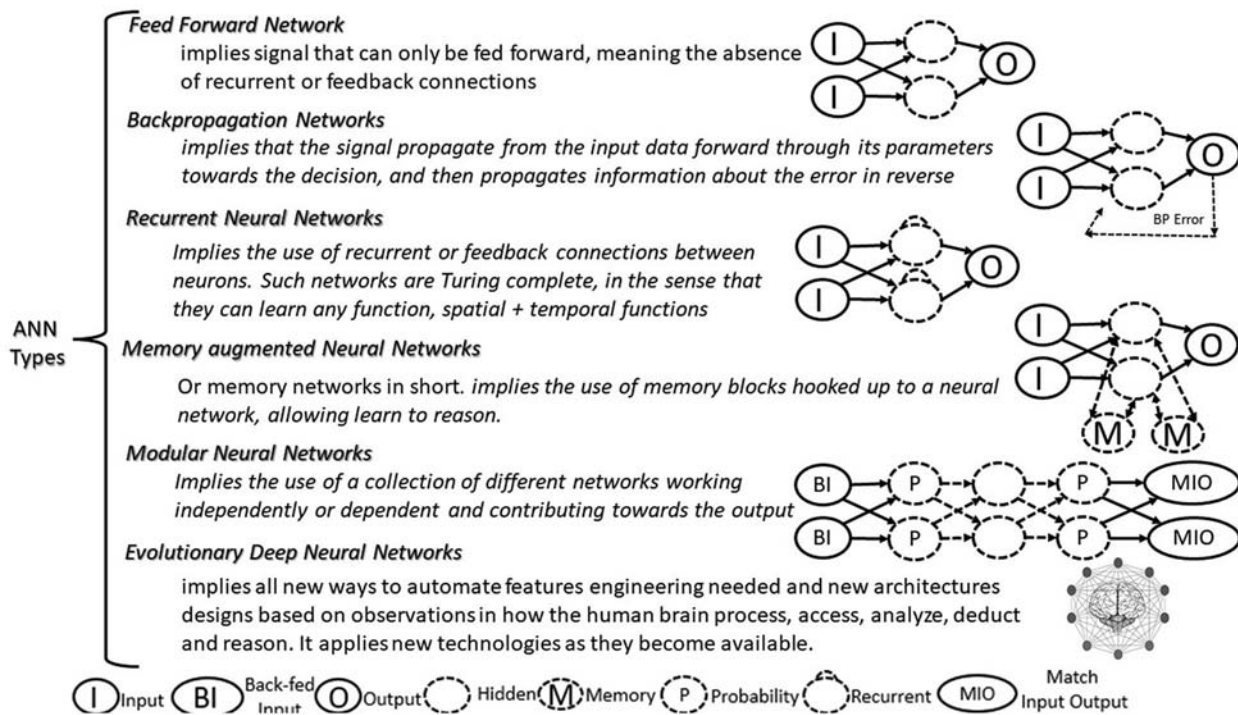


FIGURE 1.7 Artificial Neural Network types based on the way data are processed in neurons (node).

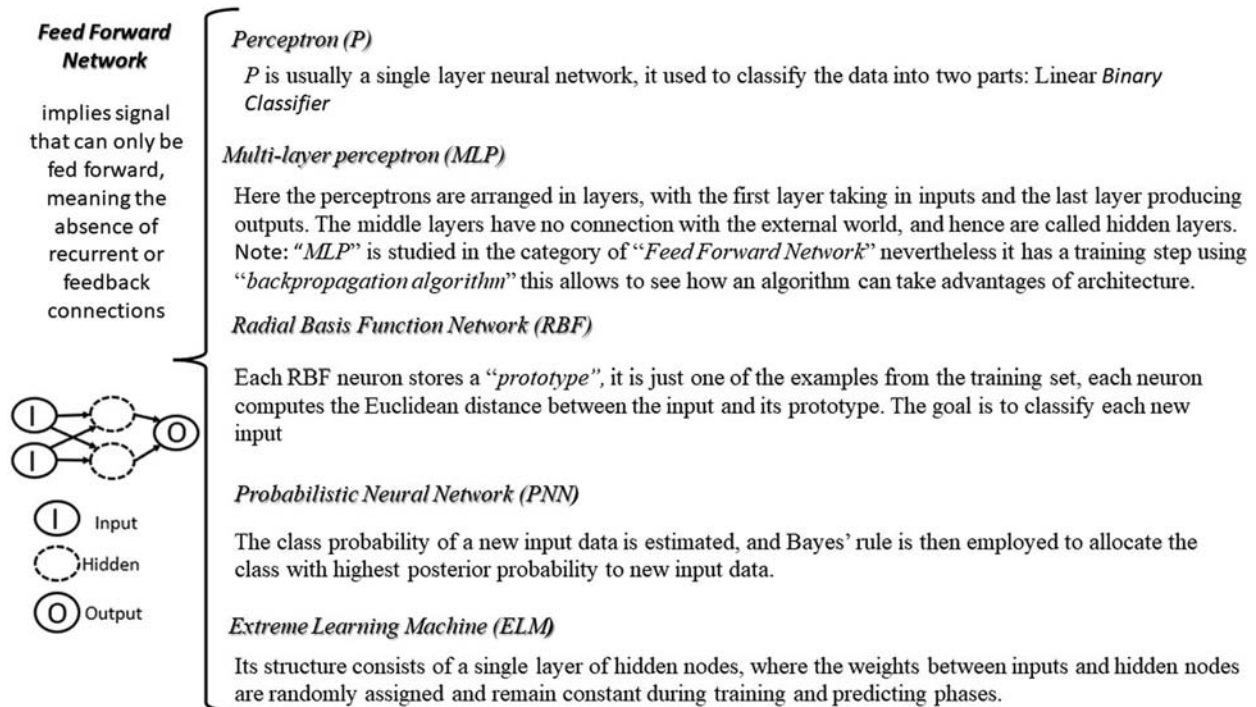


FIGURE 1.8 Examples of ANNs as Feed Forward Networks.

- “Multilayer perceptron (MLP)” also called “Feed forward Neural Network (FFN)” is when the network has three or more hidden layers. In “MLP,” the perceptrons are arranged in layers, with the first layer taking in inputs and the last layer producing outputs. The middle layers have no connection with the external world, and hence are called hidden layers. The goal of a “MLP” is to approximate some function “ $f^*(x)$.” For example, for a classifier, “ $y = f^*(x)$ ” maps an input “ x ” to a category “ y .” Then, an “MLP” defines a mapping “ $y = f(x, \emptyset)$ ” and learns the value of the parameters “ \emptyset ” that result in the best function approximation. “MLP” utilizes “supervised learning” that can distinguish data that are not linearly separable.

The “MLP” is studied in more detail in [Section 5.2.2](#).

- “Radial Basis Network (RBF)” can be used in nonlinear classifications and other applications, such as function approximation. It performs classification by measuring the input’s similarity to examples from the training set. Each “RBF neuron” stores a “prototype,” which is just one of the examples from the training set. Where the goal is to “classify a new input,” each “neuron” computes the “Euclidean distance” between the input and its prototype. If the input more closely

resembles the class A prototypes than the class B prototypes, it is classified as class A, and so on. “RBF” have many applications, such as function approximation, time series prediction, classification, and system control.

The “RBF” is studied in more detail in [Section 5.2.3](#).

- “Probabilistic Neural Network (PNN)” is a type of “ANN” derived from the “Bayesian network” [25] and a statistical algorithm called “Kernel Fisher discriminant analysis” [26], where the parent “probability distribution function (PDF)” of each class is approximated by a “Parzen window” method and a nonparametric function. Then, using “PDF” of each class, the class probability of a new input data is estimated and “Bayes’ rule” is then employed to allocate the class with highest posterior probability to new input data. In a “PNN,” the operations are organized into a “multilayered feed forward network” with four layers: “Input layer,” “Pattern layer,” “Summation layer,” and “Output layer.” It is widely used in classification and pattern recognition problems [27].

The “PNN” is studied in more detail in [Section 5.2.4](#).

- “*Extreme Learning Machine (ELM)*” is a method that is essentially a “*single feed forward neural network*”; its structure consists of a single layer of hidden nodes, where the weights between inputs and hidden nodes are randomly assigned, this means that it does not need a learning process to calculate the parameters of the models, and remains constant during training and predicting phases. On the contrary, the weights that connect hidden nodes to outputs can be trained very fast [28]. The greatest advantage of “*EMLs*” is that they are very cheap computationally for implementing online models [29]. “*ELM*” is used for pattern classification and function approximation.

The “*ELM*” is studied in more detail in [Section 5.2.5](#).

- And many others that are based on *Feed Forward neural network architecture*.

1.5.4 Backpropagation neural network

“*Backpropagation neural networks*” imply that the signal propagates from the input data forward through its parameters toward the decision, and then propagates information about the error in reverse, and thus in this way can adjust the parameter until finding the smallest error. Some

frequently used examples of “*Backpropagation neural networks*” are shown in [Fig. 1.9](#): “*Auto Encoder (AE)*,” “*Variational Auto Encoder (VAE)*,” “*Denoising Auto Encoder (DAE)*,” “*Sparse Auto Encoder (SAE)*,” “*Deep Convolution Network (DCN) or ConvNet (CNN)*,” “*Deconvolutional Network (DN)*,” “*Deep Convolutional Inverse Graphics Network (DCIGN)*,” “*Generative Adversarial Network (GAN)*,” “*Deep Residual Network (DRN)*, or *Deep ResNet*” and others.

- “*Auto Encoder (AE)*” is an “*unsupervised artificial neural network*” that learns how to efficiently compress and “*encode data from images, and how to reconstruct the data back from the reduced encoded representation*” to a representation that is as close to the original input as possible. “*Auto Encoder*,” by design, reduces data dimensions by learning how to ignore the noise in data and images [30]. “*AE*” consists of four main parts: (1) “*Encoder*” is where the model learns to reduce input dimensions and compress input data; (2) “*Bottleneck*” is a layer that contains the compress representation of the input data of images; (3) “*Decoder*” is where the model learn to reconstruct data from the encoded representation, and (4) “*Reconstruction loss*” is a method that measures the decoder and how outputs compare to the original data.

Note: An “Auto Encoder Neural Network (AE)” is an unsupervised learning or feature learning that has

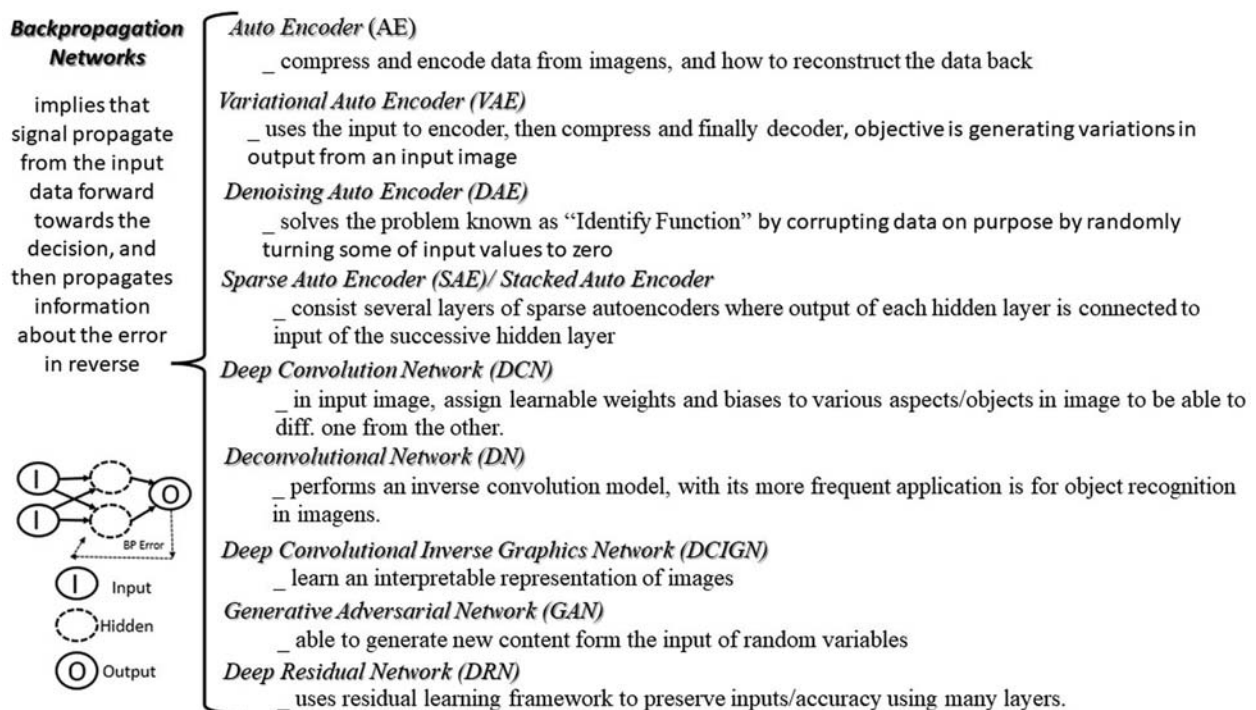


FIGURE 1.9 Examples of ANNs as backpropagation networks.

“Encoders” and “Decoders,” they are “feed forward NN but applies backpropagation algorithms,” for setting the target values to be equal to the inputs, that is, for “ $y(i) = x(i)$.”

The “AE” is studied in more detail in [Section 5.4.1](#).

- “Variational Auto Encoders (VAE)” are powerful generative models, with diverse applications: from generating fake human faces, to producing purely synthetic music. VAE uses the input to the encoder, then compress and finally decoder, its objective is to replicate the same image from the input in the output. In generative models, the objective is to generate variations in the output from an input image.

The “VAE” is studied in more detail in [Section 5.4.2](#).

- “Denoising Auto Encoder (DAE)” solves the problem known as “Identify Function*” by corrupting the data on purpose by randomly turning some of the input values to zero. In general, the percentage of input nodes that are set to zero is about 50%. When calculating the “Loss function,” it is important to compare the output values with the original input, not with the corrupted input. This way, the risk of learning the identity function instead of extracting features is eliminated [31].

Note:* Identify Function or “Null Function” is present in neural network when there are more nodes in the hidden layer than there are in the inputs, this issue could limited the mathematical learning of important behaviors to obtain a useful AI model

The “DAE” is studied in more detail in [Section 5.4.3](#).

- “Sparse Auto Encoder (SAE)” consists of a single hidden layer, which is connected to the input vector by a weight matrix forming the encoding step. The hidden layer then outputs to a reconstruction vector, using a tied weight matrix to form the decoder. “A stacked auto encoder” is a neural network consisting of several layers of “sparse autoencoders” where the output of each hidden layer is connected to the input of the successive hidden layer [32]. “Stacked auto encoder” improves accuracy in deep learning with noisy autoencoders embedded in the layers. Stacked auto encoders are used for many medical science purposes, such as

“P300 Component Detection and Classification of 3D Spine Models in Adolescent Idiopathic Scoliosis,” where, the “classification” of the rich and complex variability of spinal deformities is critical for comparisons between treatments and for long-term patient follow-ups.

The “SAE” is studied in more detail in [Section 5.4.4](#).

- “Deep Convolution Network (DCN) or ConvNet (CNN)” is a class of “deep neural networks,” most applied to analyzing visual imagery. “DCN” can take in an input image, assign importance using learnable weights and biases to various aspects/objects in the image, and be able to differentiate one from the other. A “ConvNet” architecture is in the simplest case a list of layers that transform the image volume into an output volume. The preprocessing required in a “ConvNet” is much lower as compared to other classification algorithms. “CNNs” are regularized versions of multilayer perceptrons that are fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. In the classical backpropagation algorithm, the weights are changed according to the gradient descent direction of an error surface [33]. The architecture of a “ConvNet” is analogous to that of the connectivity pattern of neurons in the human brain and was inspired by the organization of the visual cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the “Receptive Field.” A collection of such fields overlaps to cover the entire visual area. “DCN” are used in applications for image and video recognition, image analysis and classification, media recreation, recommendation systems, natural language processing, etc. [34,35].

The “DCN” is studied in more detail in [Section 5.4.5](#).

- **Deconvolutional Network (DN)** also known as “deconvs” or “transposed convolutional neural networks.” “DN” is a neural network that performs an “inverse convolution model”; its most frequent application is for object recognition in images. “DN” is an “unsupervised construction of hierarchical image representation learning” of mid- and high-level image representation, this can be achieved using feature hierarchy from alternative layers of “convolutional sparse coding or deconvolution”; it is a method

for learning by shift-invariant dictionaries in image and “*max pooling*” as a sample-based discretization process [36].

The “DN” is studied in more detail in [Section 5.4.6](#).

- “*Deep Convolutional Inverse Graphics Network (DCIGN)*” has the objective to learn an interpretable representation of images that is disentangled with respect to various transformations such as object out-of-plane rotations, lighting variations, and texture. The “*DCIGN model*” is composed of “*multiple layers of convolution and deconvolution operators*” and is trained using the “*Stochastic Gradient Variational Bayes (SGVB)*” algorithm [37].

The “DCIGN” is studied in more detail in [Section 5.4.7](#).

- “*Generative Adversarial Networks (GAN)* belong to the set of “*generative models*.” It means that they are able to produce/to generate new content from the input of random variables, then a generative network is trained to maximize the final classification error, the results are a generated distribution, and finally a discriminate network is trained to minimize the final classification error that is used as a reference metric for both networks [38].

The “GAN” is studied in more detail in [Section 5.4.8](#).

- “*Deep Residual Network (DRN) or Deep ResNet*” is a “*neural network using a residual learning framework*” that preserves inputs and improves accuracy using many layers. “*Deep ResNet*” architecture “*holds many stacked layers including convolutional, pooling, and fully connected*.” This network has a “*skip function*” that reduces the number of times a linear function is used to achieve an output creating “*residual block*” that eliminates the “*varnished gradients*,” so that when a gradient becomes very small, even a very big change in the input will not affect the output as desired; and “*exploding gradients*” so that when a gradient becomes exponentially big, the algorithm can no longer be used to train the model. “*DRN*” are often used for image recognition applications.

The “DRN” is studied in more detail in [Section 5.4.9](#).

- And many others that use “*Backpropagation neural networks*” architecture.

1.5.5 Recurrent neural networks

“*Recurrent neural networks*” imply the use recurrent or feedback connections between neurons. Such networks are “*Turing complete*,” in the sense that they can learn any function, spatial + temporal functions. Some frequently used examples of “*Recurrent neural networks*” are shown in [Fig. 1.10](#), these are: “*Recurrent Neural Network (RNN) vanilla*,” “*Long/short-term memory (LSTM)*,” “*Gated recurrent unit (GRU) networks*,” “*Recurrent convolutional neural networks (RCNN)*,” “*Hopfield Network (HN)*,” “*Boltzmann Machine (BM)*,” “*Restricted Boltzmann Machine (RBM)*,” “*Liquid State Machine (LSM)*,” “*Echo State Network (ESN)*,” “*Korhonen Network (KEN)*,” and many more. Where:

- “*Recurrent Neural Network (RNN) vanilla*” takes the previous output or hidden states as inputs. The composite input at time “*t*” has some historical information about the happening at time $T < t$.

The “RNN vanilla” is studied in more detail in [Section 6.2.1](#).

- “*Long/Short-Term Memory (LSTM)*” is a special “*RNN*” capable of learning long-term dependencies, simulating in its feedback connections a “*general-purpose computer*.” It can be used for classifying, processing, and predictions. The “*LSTM*” process uses single data points, such as images to sequences of data as text, speech, audio, and video [39]. “*LSTM gates*” have three types of memory cells: “*Forget Gate*” that decides what information to discard from the cell; “*Input Gate*” that decides which values from the input to update the memory state; and “*Output Gate*” that decides what to output based on input and the memory of the cell [40].

The “LSTM” is studied in more detail in [Section 6.2.2](#).

- “*Gated Recurrent Unit (GRU)*” is a “*variant RNN*” like an “*LSTM*” unit but “*without an output gate using a gating mechanism*.” A “*GRU*” can be considered a specific variation of an “*LSTM*” unit because both have a similar design and produce equal results in some cases. “*GRU*” uses a gating mechanism to control and manage the flow of information between

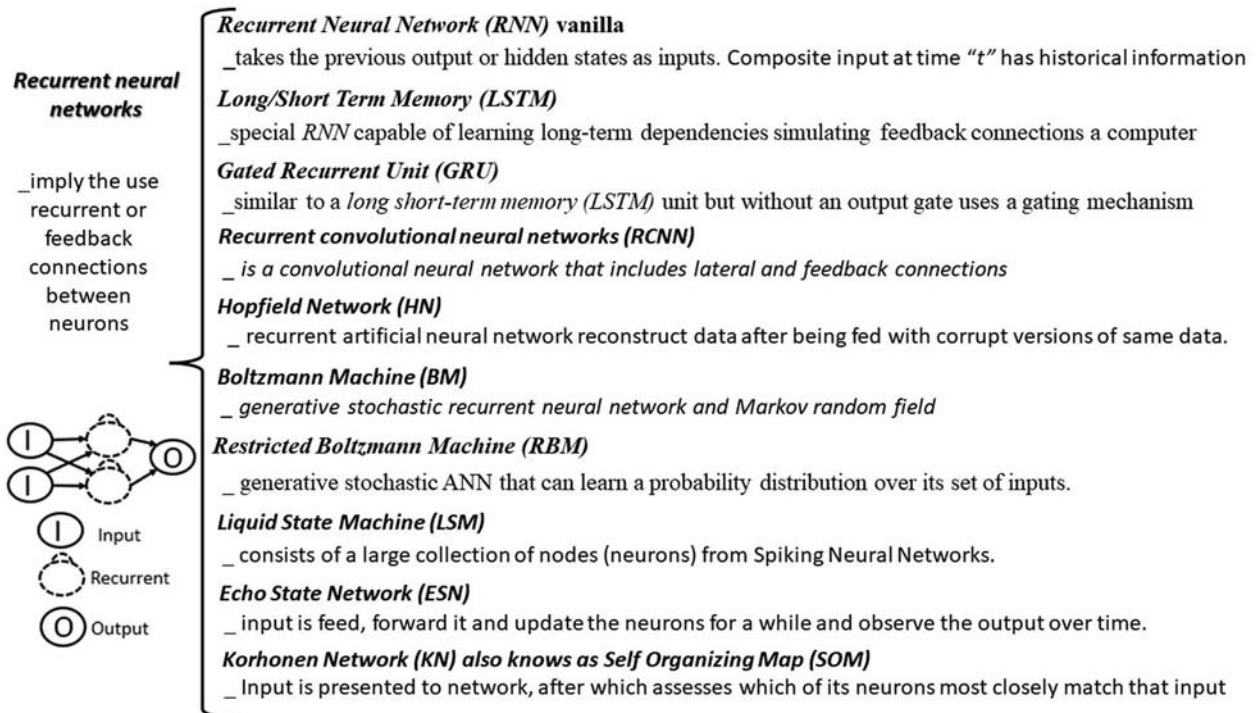


FIGURE 1.10 Examples of ANN as Recurrent neural networks.

cells in the neural network. “GRUs” can solve the “vanishing gradient problem*” by using an “update gate” and a “reset gate.” The “update gate” controls information that flows into the memory, and the “reset gate” controls the information that flows out of the memory. The “update gate and reset gate” are two vectors that decide which information will get passed on to the output. They can be trained to keep information from the past or remove information that is irrelevant to the prediction.

Note: The vanishing gradient problem occurs when the calculated partial derivatives are used to compute the gradient as one goes deeper into the network. Since the gradients control how much the network learns during training, if the gradients are very small or zero, then little to no training can take place, leading to poor predictive performance.*

The “GRU” is studied in more detail in [Section 6.2.3](#).

- “Recurrent convolutional neural network (RCNN)” is a “convolutional neural network that includes lateral and feedback connections.” Feed forward neural networks provide the dominant model of how the brain performs visual object recognition. However, these

networks lack the lateral and feedback connections, and the resulting “recurrent neuronal dynamics, of the ventral visual pathway in the human and nonhuman primate brain” [41].

The “RCNN” is studied in more detail in [Section 6.2.4](#).

- “Regional-Convolutional Neural Network (R-CNN) Object detection in AI Models” is a special kind of standard “CNN”, it can handle different spatial locations with different ratios. “R-CNN” can analyze a huge number of regions, using special methods to avoid long computational process and powerful hardware. There are some “AI procedures to resolve this problem,” such as “Regional-Convolutional Neural Network (R-CNN),” “Fast R-CNN”, “YOLO” and others.

The “R-CNN” is studied in more detail in [Section 6.2.5](#).

- “Hopfield Network (HN)” is a form of “recurrent artificial neural network” that can reconstruct data after being fed with corrupt versions of the same data. It usually works by first learning several binary patterns and then returning the one that is the most similar to a

given input. It can be described as a network of nodes, representing neurons that are connected by links; each unit has one of two states at any point in time and the vector represents the state of each node. The links represent the connections between nodes, and they are symmetric.

The “HN” is studied in more detail in [Section 6.2.6](#).

- “Boltzmann Machine (BM)” is a type of “stochastic recurrent neural network” and Markov random field.” “BM” is a generative net because it does not expect input data. It generates data as a “unsupervised model,” which involves learning a “probability distribution” from an original dataset and using it to make inferences about never before seen data. “BM” has an input visible layers and one or several hidden layers, where everything is connected to everything. Connections are bidirectional, visible neurons connected to each other and hidden neurons also connected to each other.

The “BM” is studied in more detail in [Section 6.2.7](#).

- “Restricted Boltzmann Machine (RBM)” is a variant of “BM.” “RBM” is a generative “stochastic artificial neural network” that can “learn a probability distribution” over its set of inputs. It can be trained in either “supervised or unsupervised” ways, depending on the task.

The “RBM is” studied in more detail in [Section 6.2.8](#).

- “Liquid State Machine (LSM)” consists of a large collection of nodes or neurons from “Spiking Neural Networks*.” Each node receives time varying input from external sources as inputs, and from other nodes, where nodes are randomly connected to each other. The recurrent nature of the connections turns the time varying input into a “spatiotemporal pattern of activations” in the network nodes. The “spatiotemporal patterns” of activation are read out by linear discriminant units. The word “liquid” comes from the analogy drawn to dropping a stone in water or other liquid to generate ripples in it. The input simulates the motion of the falling stone, which is converted into a spatiotemporal pattern of liquid displacement representing the ripples, then the ripples can be evaluated to understand what is

happening in the system being studied [42]. Like other kinds of neural networks, “liquid state machines and similar builds are based around the neurobiology of the human brain.” “LSM” has applications such as speech and audio recognition, image pattern recognition, music classification, robot path planning, fingerprint scanners, facial emotion recognition, and others [43].

Note*: Spiking neural networks (SNNs) are artificial neural networks that more closely mimic natural neural networks. In addition to neuronal and synaptic state, SNNs incorporate the concept of time into their operating model.

The “LSM” is studied in more detail in [Section 6.2.9](#).

- “Echo State Network (ESN)” is another type of “recurrent neural network (RNN)” using “supervised learning.” An “ESN” is not organized in a standard set of layers, it has random connections between the neurons and the training is different: instead of feeding the input and calculate the weight and the backpropagating error used at typical RNN, the input is feed, calculate the weight and update the neurons value for a while to observe the output that change over time in a special a “dynamic reservoir” [44].

The “ESN” is” studied in more detail in [Section 6.2.10](#).

- “Korhonen Network (KN), also known as Self Organizing Map (SOM),” is a type of “artificial neural network (ANN)” used as “unsupervised learning” for classification. Input is presented to the network, after which the network assesses which of its neurons most closely match that input. They use a neighborhood function to preserve the topological properties of the input space. These neurons are then adjusted to match the input even better, dragging along their neighbors in the process. How much the neighbors are moved depends on the distance of the neighbors to the best matching units [45]. It seems to be the most natural way of learning, which is used in our brains, where no patterns are defined.

The “KN” is studied in more detail in [Section 6.2.11](#).

- And many others that use “Recurrent neural networks architecture.”

1.5.6 Memory augmented neural networks

“Memory augmented neural networks” or “memory networks” in short “imply the use of memory blocks hooked up to a neural network, allowing the learning to reason.” Some examples are shown in Fig. 1.11: *Neural Turing machines (NTM)*, *Differentiable Neural Computers (DNC)*, where:

- “*Neural Turing Machine (NTM)*” is the first application of “DL to extend the capabilities of ANN by coupling them to external memory”; it is based on “RNN” coupled with external memory resources which they can interact with by attentional processes. The combined system is analogous to a “*Turing Machine* or *Von Neumann architecture* but is *differentiable end-to-end*,” allowing it to be efficiently trained with “*gradient descent*.” Results demonstrate that “*NTMs*” can infer simple algorithms such as copying, sorting, and associative recall from input and output examples. Experiments demonstrate that “*NTMs*” are capable of learning simple algorithms and are capable of generalizing beyond the training regime [46].

Note:* Gradient descent method uses a feedback loop to adjust the model based on the error it observes between its predicted output and the actual output.

The “*NTM*” is studied in more detail in Section 6.3.1.

- “*Differentiable Neural Computers (DNC)*” use a form of “*memory augmented neural network that can learn using its memory to answer complex questions related of structured data, maps, trees, etc.*” The controller of the “*DNC*” can read/write from multiple locations in memory. Memory can be searched based on the content of each location, or the associative temporal links can be followed forward and backward to recall information written in sequence or in reverse. The read-out information can be used to produce answers to questions or actions to take in an environment [47]. “*The idea is to take the classical Von Neumann computer architecture and replace the CPU with an RNN, which learns when and what to read from the random-access memory (RAM)*” [48].

The “*DNC*” is studied in more detail in Section 6.3.2.

- And others that are in research and use “*Memory augmented neural networks architectures.*”

1.5.7 Modular neural networks

“*Modular Neural Networks*” implies the use of “*a collection of different networks working independently or dependently and contributing toward the output.*” Each “*neural network has a set of inputs which are unique compared to other networks constructing and performing subtasks.*”

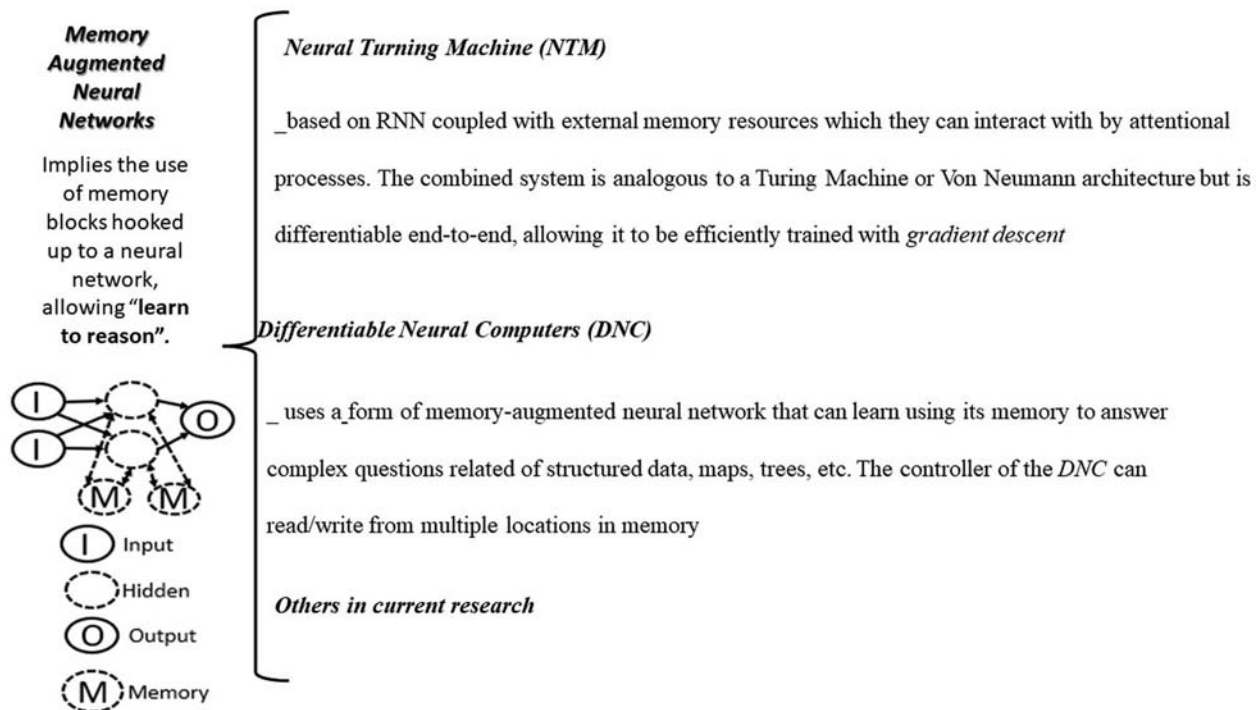


FIGURE 1.11 Examples of ANNs as Memory augmented neural networks.

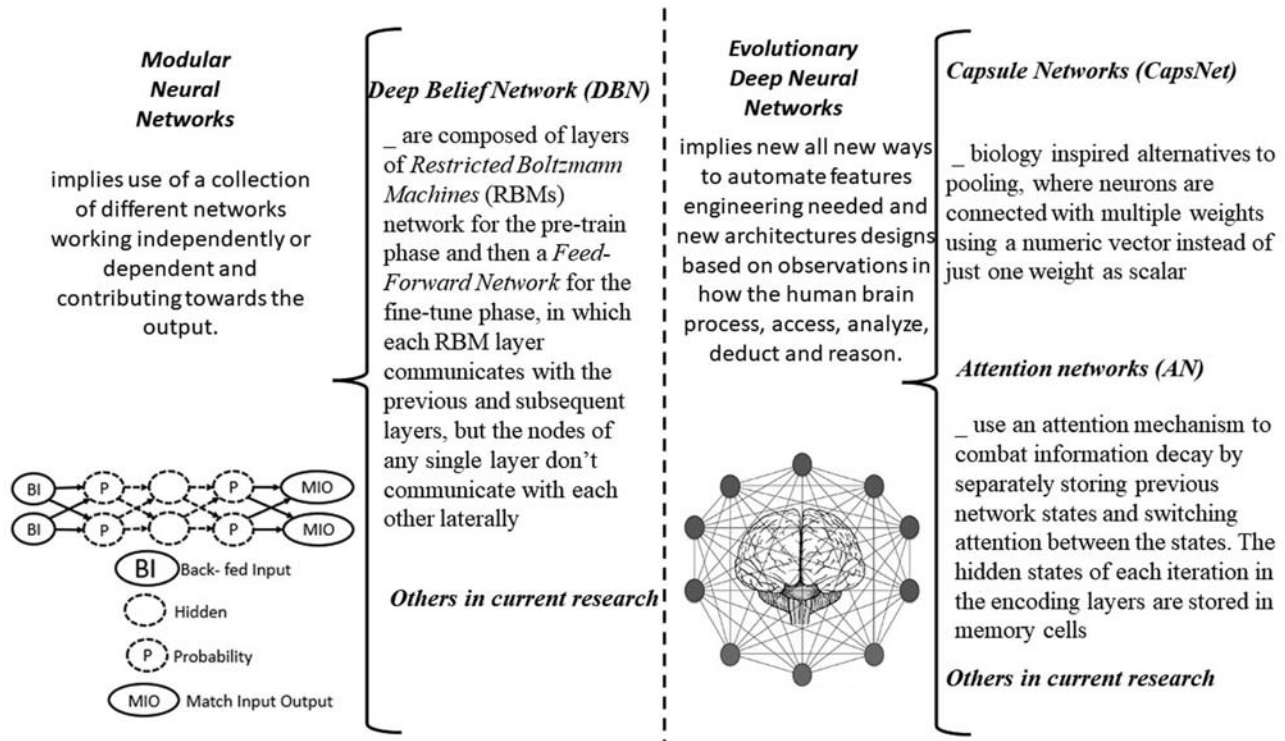


FIGURE 1.12 Examples of (left) Modular Neural Networks and (right) Evolutionary Deep Neural Networks.

If they are independent, these networks do not interact or signal each other in accomplishing the tasks and can breakdown a large computational process into smaller components decreasing the complexity. This breakdown will help in decreasing the number of connections and negates the interaction of these networks with each other, which in turn will increase the computation speed. If the networks are dependent, one network complements the work of the other. Some examples are shown in Fig. 1.12:

- “*Deep Belief Networks (DBN)*” are composed of layers of “*RBMs*” networks for the pretrain phase and then a “*FFN*” for the fine-tune phase, in which each RBM layer communicates with the previous and subsequent layers, but the nodes of any single layer do not communicate with each other laterally. *DBN* can be used as a classifier if ending with a *Softmax** layer or as unsupervised learning otherwise. *DBN* are used to recognize, cluster, and generate images, video sequences, and motion-capture data.

Note:* *Softmax function takes an input as a vector of K real numbers and normalizes it into a probability distribution consisting of K probabilities. The input vectors that could contain negative, or greater than 1 values are converted to components in the interval (0,1), and they will add up to 1, representing probabilities values.*

The “*DBN*” is studied in more detail in Section 6.4.1.

- And others that are currently in research and use *Modular Neural Networks architecture* in different applications, such as biological, psychological, hardware, computational, multimodule decision-making strategies, etc.

1.5.8 Evolutionary deep neural networks

“*Evolutionary Deep Neural Networks*” imply all new architectures for “*ANN*” with the purpose of automate features engineering needed, typically are “*new designs based on observations in how the human brain process, access, analyze, deduct and reason applying new technologies as they become available.*” Examples are:

- “*Capsule Networks (CapsNet)*” are biology inspired alternatives to pooling, where neurons are connected with multiple weights using a numeric vector instead of just one weight as scalar. This method uses new concept, such as “*capsules*” that perform some quite complicated internal computations on their inputs and then encapsulate the results of these computations into a small vector of highly informative data, “*dynamic routing between capsules algorithm*” that allows capsules to communicate with each other and create representations similar to scene graphs in computer graphics. The “*CapsNet*” has 2 parts: “*encoder*” and “*decoder.*” The first three layers are “*encoders,*” and the second three are “*decoders.*” “*CapsNet*” uses a lot of computational resources of “*GPU*” type and it can

be used to obtain a better model hierarchical relationship, and to more “*closely mimic biological neural organization*” [49].

The “CapsNet” is studied in more detail in Section 6.5.1.

- “Attention networks (AN)” use an “*attention mechanism*” to combat information decay by separately storing previous network states and switching attention between the states. The hidden states of each iteration in the encoding layers are stored in memory cells. The “*decoding layers are connected to the encoding layers, but it also receive data from the memory cells filtered by an attention context.*” This filtering step adds context for the decoding layers stressing the importance of features. The “AN” producing this context is trained using the error signal from the output of decoding layer. Moreover, the “*attention context can be visualized giving valuable insight into which input features correspond with what output features.*” “AN” also includes the “*transformer architecture*” [50]. “AN” has the objective of accomplishing “*text classification*” based on “*words make sentences and sentences make documents*”; it uses “*stacked recurrent neural networks*” at the word level followed by an “*attention model*” to extract such words that are important to the meaning of the sentence and aggregate the representation of those informative words to form a sentence vector. Then the same procedure is applied to the derived sentence vectors which then generate a vector that conceives the meaning of the given document and that vector can be passed further for text classification [51].

The “AN” is studied in more detail in Section 6.5.2.

- And others that are currently being researched and use *Evolutionary Deep Neural Networks*

ANN types will be studied with examples and exercises in Chapter 5, “*Deep Learning Models Principles Applied to Biomedical Engineering,*” and Chapter 6, “*Deep Learning Models Evolution Applied to Biomedical Engineering.*”

1.6 Cognitive science

“*Cognitive science (CoSi)*” is the “*interdisciplinary scientific study of the mind and its processes.*” “*CoSi*” examines the nature, tasks, and the functions of human cognition as

the process of acquiring knowledge and understanding through thought, experience, and the senses. “*CoSi*” as the objective of develop theories about human: perception, action, memory, attention, reasoning, decision-making, language use, and learning. To achieve these goals uses a multidiscipline science, such as “*Philosophy,*” “*Psychology,*” “*Anthropology,*” “*Biology,*” “*Computer Science through Artificial Intelligence,*” “*Linguistics,*” “*Neuroscience,*” “*Education,*” and “*Mathematics.*” Where their relationship with cognition can be explained as follow:

- “*Philosophy*” studies the fundamental nature of knowledge, reality, and existence, while “*cognition studies the intelligence mind.*”
- “*Psychology*” studies the human mind and its function, while “*cognitive psychology tries to make functional models of the mind.*”
- “*Anthropology*” studies the human societies, their cultures, and their development. It is complemented by the “*cognition interest in evolution, social groups, communications and culture.*”
- “*Biology*” studies the living organism and “*cognition focuses on the survival of organisms and species.*”
- “*Computer Science through Artificial Intelligence*” focuses on obtaining computational models using processes that include learning, reasoning, and self-correction. “*AI applies algorithms that can automatically learn and improve from experience without being explicitly programmed; in other words, the AI applications are cognitive themselves.*”
- “*Linguistics*” studies the use of language to represent information, and “*cognition is related to the representation and manipulation of the information.*”
- “*Neuroscience*” studies the development and function of the nervous system, which includes the brain, spinal cord, and nerve cells throughout the body. While “*cognition covers the relationship between the mind and the brain.*”
- “*Education*” is the process of receiving or giving systematic instruction. While “*cognitive science is related to the research and finding of ways to improve education.*”
- “*Mathematics*” is the abstract science of number, quantity, and space. While “*cognitive science can explain the nature of mathematical thinking to analyze cognitive models.*”

“*Cognitive Science (CoSi)* is applied with *Cognitive Computing (CC),*” which has the “*main objective of simulating the human thought process in an AI computerized mode with learning and reasoning.*” It uses “*self-learning*” algorithms of “*ML*” and “*DL,*” and applies data mining, pattern recognition, natural language processing, and other techniques that will allow a computer system to mimic the how the human brain works. These concepts are introduced

(Continued)

(Continued)

in Chapter 2, "Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to Help in the Solution of AI Biomedical Engineering Problems," and Chapter 7, "Cognitive Learning and Reasoning Models Applied to Biomedical Engineering."

1.6.1 Cognitive computing

"Cognitive Computing (CC)" also known as "Cognitive Technology is a subfield of AI that allows us to analyze human cognition using computerized AI models of learning and reasoning to represent the processes studied in cognitive science." "CC" helps humans to make better decisions with the help of smart machines based on "CoS" that study the human brain and how it functions.

"CC" makes use of a number of technological resources to create "Cognitive Computing Agents Systems"; these are complete powerful computer systems locally and distributed in the cloud, that have the objective to integrate complex human interactions for human-like analysis. "Cognitive computing agents" have the following characteristics: speak and understand fluent human language, learn, and self-learn, cognitive analysis, make decisions, make predictions, and many other intelligent functions [52]. There are many architectural frameworks proposed for "Cognitive Computing Agents Systems," such as SOAR, ACT-R, NARS, CLARION, LIDA, EPIC, MDB, COGPRIME, eBICA, MILECOGs, and others. The proposed general architecture framework for a "AI-Cognitive Computing Agents Systems (CCAS)" is shown in Fig. 1.13, and includes the following 13 modules: "signal recognition instruments for speech, vision, and imaging bioinstruments," "cognitive detection of human-like abilities,"

This book proposes the concept of "Cognitive Computing as AI learning and reasoning" that allows evaluate human cognitive status and assist humans in their decision-making process under learning and reasoning algorithms. "Cognitive Computing (CC)" is based on many AI self-learning algorithms and language tools relying in multidiscipline that need apply different fields and methods as data mining, pattern recognition, classification, prediction and others as Natural Language Processing, images

analysis to collect information to feed themselves on, and special computer systems for cognitive data processing. This AI system must understand natural language and interact with the humans in a more natural way based on human cognition, with the ability to understand, grasp, and reason all collected cognitive information. For this goal is propose a proposed a "General Architecture framework of AI-Cognitive applying Computing Agents System (AI-CCAS)" as shown in Fig. 1.13.

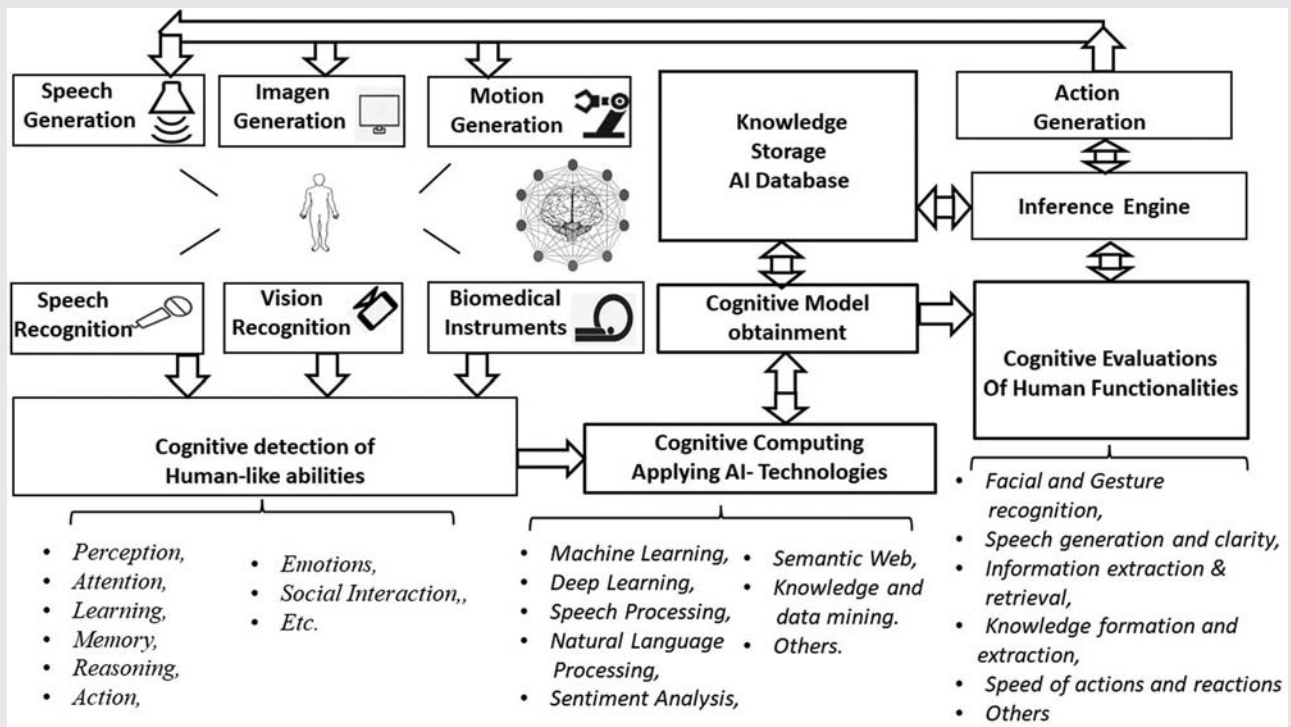


FIGURE 1.13 Proposed general architecture framework of an AI-Cognitive Computing Agents System (AI-CCAS).

“cognitive computing applying AI technologies,” “cognitive models obtainment,” “cognitive evaluations of human functionalities,” “inference engine,” “knowledge storage AI database (KSAIDB),” “action generation,” and “output devices for signal generation: speech generation, image generation and motion generation.”

1.6.2 Signal recognition instruments*

Many inputs devices for signal recognition can be connected to the “Cognitive Computing Agents,” the more frequently are “speech recognition,” “vision recognition,” and “biomedical instruments.”

- “Speech recognition or Speech to text” is an interdisciplinary subfield of computational linguistics that has methodologies and technologies that enables the recognition and translation of spoken language into text by computers. The “speech capture” is made using microphones, besides this technology is being used to replace other methods of input like typing, clicking, or selecting in other ways.

The “Speech recognition or Speech to text” is studied in more detail in [Section 2.6.5](#).

- “Vision recognition” also known as “computer vision” is an interdisciplinary scientific field that deals with how computers can be made to gain high-level understanding from digital images or videos. “Computer vision” tasks include methods for acquiring, processing, analyzing, and understanding digital images, and extraction of high-dimensional data from the real world in order to produce text, numerical or symbolic information. The vision is captured by devices as cameras for recording visual images in form of photographs, scanning documents, or video signals.

The “Vision recognition” is studied in more detail in [Section 5.4.5](#) and [Section 6.2.5](#).

- “Biomedical instruments” are a subbranch of “biomedical engineering.” Mainly focuses on how electronics equipment can capture and measure physiological patient data. Basically, they are for diagnostic biomedical using instruments, such as “Electrocardiogram (ECG)” for monitoring cardiac waves, “Electroencephalogram (EEG)” for monitoring electrical activity of the brain, “Electromyography (EMG)” for monitoring electrical muscle activity, “Electrooculography (EOG)” for monitoring eye

movement, “glucometers” for monitoring blood sugar levels, images diagnostic bioinstruments as “X-Rays,” “magnetic resonance imaging (MRI)” that creates detailed images of organs and tissues, “computed tomography (CT)” that creates pictures of organs, bones, and other tissues, etc.

Many “Biomedical instruments” are studied with more detail in my previous book “Applied Biomechatronics Using Mathematical Models” [2].

1.6.3 Cognitive detection of human-like abilities*

To create a “AI-cognitive human model” is necessary to include in its design the “cognitive human-like abilities” as [53]: “Perception,” “Attention,” “Learning,” “Memory,” “Reasoning,” “Action,” “Emotions,” “Social Interaction,” “Planning,” “Motivation,” “Actuation,” “Communication,” etc. Where each one can be evaluated in different aspects as:

- *Perception:* Vision, Smell, Touch, Taste, Audition, Cross-modal, Proprioception, others.
- *Attention:* Visual, Auditory, Social, Behavioral, others.
- *Learning:* Imitation, Reinforcement, Dialogic, Media-Oriented, Experimentation, others.
- *Memory:* Working, Episodic, Implicit, Semantic, Procedural, others.
- *Reasoning:* Induction, Deduction, Abduction, Physical, Causal, Associational, others.
- *Emotion:* Perceived, Expressed, Control, Understanding, Sympathy, Empathy, others.
- *Social Interaction:* Communication, Social Inference, Cooperation, Competition, Relationship, others.
- *Planning:* Tactical, Strategic, Physical, others.
- *Motivation:* Subgoal creation, Affected based, Deferred, Gratification, Altruism, others.
- *Actuation:* Physical Skills, Tool use, Navigation, others.
- *Communication:* Verbal, Gestural, Musical, Language Acquisition, Cross-modal, others.

“Cognitive detection of human-like abilities” is studied in more detail in [Chapter 7](#), Cognitive Learning and Reasoning Models Applied to Biomedical Engineering.

1.6.4 Cognitive computing applying AI technologies*

“AI technologies” that are required for a computer system or robot to build cognitive models that mimic human thought processes may include: “machine

learning,” “deep learning,” “speech processing,” “natural language processing,” “sentiment analysis,” “semantic web,” “knowledge and data mining,” “cognitive computing,” and many other “AI technologies.” Where:

- “Machine Learning (ML)” is a subset of “AI” that provides systems with the ability to automatically learn and improve from experience without being explicitly programmed.

Note: ML will be explained with more details including examples and exercise in [Chapter 4](#), “Machine Learning Models Applied to Biomedical Engineering,” of this book.

- “Deep Learning (DL)” is a subset of “ML,” in which an “AI model learns to perform classifications task directly from images, text and sound.” “DL” is usually implemented using an “artificial neural network (ANN)” architecture.

Note: DL will be explained with more details including examples and exercise on [Chapter 5](#), “Deep Learning Models Principles Applied to Biomedical Engineering,” and [Chapter 6](#), “Deep Learning Models Evolution Applied to Biomedical Engineering,” of this book.

- “Speech Processing” is the study of speech signals and the computer processing methods of these signals in a digital representation. Aspects of speech processing includes the acquisition, manipulation, storage, transfer, and output of speech signals.

“Speech Processing” is studied in more detail in [Chapter 2](#), Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to Help in the Solution of AI Biomedical Engineering Problems, and [Chapter 3](#), Cognitive Learning and Reasoning Models Applied to Biomedical Engineering.

- “Natural Language Processing (NLP)” is the use of computers to generate and/or understand written and spoken language for some practical purpose: machine translation, natural language queries, conversing with machines, and so on.

“Natural Language Processing (NLP)” is studied in more detail in [Chapter 2](#), Introduction to Cognitive Science, Cognitive Computing and Human Cognitive Relation to Help in the Solution of AI Biomedical Engineering
(Continued)

(Continued)

Problems, and [Chapter 7](#), Cognitive Learning and Reasoning Models Applied to Biomedical Engineering.

- “Sentiment Analysis” is the process of computationally identifying and categorizing opinions expressed in a piece of text, especially in order to determine whether the writer’s attitude towards a particular topic, product, etc. is positive, negative, or neutral.

“Sentiment Analysis” is studied in more detail in [Section 7.7](#).

- “Semantic Web” is an extended version of the existing “World Wide Web” through standards, and it represents an effective means of data representation in the form of a globally linked database. By supporting the inclusion of semantic content in “Web pages,” the “Semantic Web” targets the conversion of the presently available Web of unstructured documents to a Web of information/data.

“Semantic Web” is studied in more detail in [Chapter 2](#), Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to Help in the Solution of AI Biomedical Engineering Problems, and [Chapter 7](#), Cognitive Learning and Reasoning Models Applied to Biomedical Engineering.

- “Knowledge Discovery and Data Mining” focuses on the process of extracting meaningful patterns from biomedical data, technique known as “knowledge discovery,” using automated computational and statistical tools and techniques on large datasets using data mining.

“Knowledge Discovery and Data Mining” is studied in more detail in [Chapter 2](#), Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to Help in the Solution of AI Biomedical Engineering Problems, and [Chapter 7](#), Cognitive Learning and Reasoning Models Applied to Biomedical Engineering.

1.6.5 Cognitive model obtainment*

The results obtain from analyzing the input variables of the “cognitive of human-like abilities” through the applications of algorithms of “Cognitive Computing,” allow to obtain a mathematical model that represent the relationship between them, this is known as “Cognitive AI

Model.” It simulates how the patient handle thoughts and perception captured in the input variables.

See research examples Cognitive model obtainment in [Chapter 7](#), Cognitive Learning and Reasoning Models Applied to Biomedical Engineering.

1.6.5.1 Cognitive evaluations of human functionalities

Creating a specific “*cognitive human model*,” it must be based on “*special human cognitive functionality factors of the patients*,” such as “*gesture recognition*,” “*speech generation and understanding*,” “*information retrieval*,” “*knowledge formation and extraction*,” “*speed of action to analyze limitations in different diseases*,” and other factors.

See research examples and challenge research projects of Cognitive model obtainment in [Chapter 7](#), Cognitive Learning and Reasoning Models Applied to Biomedical Engineering.

1.6.6 Inference engine*

“*Inference engine*” is a component of the “*Architecture framework of AI and Cognitive Computing Agents System (AI-CCAS)*” that applies logical rules to the knowledge base to deduct new information. For example: if it compares the data results from the human cognitive model from healthy people vs the data results of people affected with diseases or natural ageing that affected their cognition. Based on these comparisons and the logical rules generated by the “*inference engine*,” we will have methods to “*evaluate and classify*” steps of the sequence in steps of “*Cognitive declination*” in patients.

1.6.6.1 Knowledge storage AI database*

Here all information is stored and allow a random access from the inference engine during the process of deducing new information, plus the “*AI algorithms*” to maintain the information related as it grows.

See research examples of “*Inference engine*” in [Section 7.3](#).

1.6.6.2 Action generation*

This module generates all synchronized signals needed in the output devices generation.

1.6.6.3 Output devices for signal generation*

Many output devices for signal action in the patient can be used in the “*Architecture framework of AI and*

Cognitive Computing Agents System (AI-CCAS).” The more frequently are: “*speech generation*,” “*image generation*,” and “*motion generation*.”

- “*Speech generation or text to speech*” are converts from normal language text into speech; other systems render symbolic linguistic representations like phonetic transcriptions. “*Synthesized speech*” can be created by concatenating pieces of recorded speech that are stored in the “*knowledge storage AI database*” that are the results obtained from the inference engine. Usually, it is through speakers’ systems.

See “*text to speech*” examples at [Section 2.6.4](#).

- “*Image generation*” are devices that show image stored or image generated to interact with the patient; usually they are monitor, or big screen projectors that can be seen for the patients.
- “*Motion generation*” are devices using magnetic motor in systems that net movement displacement as bed, robotic arm, etc. With the intention to move or give specific motion resistance or motion exercise to patients.

Note:* The General Architecture framework of AI and Cognitive Computing Agents System (AI-CCAS) is more deeply study in [Chapter 7](#), “*Cognitive Learning and Reasoning Models Applied to Biomedical Engineering*.”

Architecture framework of AI and Cognitive Computing Agents System (AI-CCAS) is a complete computer systems that integrate complex human interaction and human-like analysis, with the purpose of deduct and obtain valid *Cognitive Models for Cognitive Science*, these AI Models simulate how the patient handle thoughts and perception applying learning and reasoning algorithms from input variables of the system.

1.7 Neuroscience, cognitive science, and AI models

“*Neuroscience*” study the development and function of the “*nervous system*,” which includes the “*brain*,” “*spinal cord*,” and “*nerve cells*” throughout the body. “*Neuroscience*” studies how the brain works in terms of mechanics, functions, and systems in order to create recognizable behaviors. Neurologists could specialize in one part of the nervous system, such as neurotransmitters, neurologic diseases or focus their specialization on specific behaviors, such as psychiatric disorders. Some major types of neurological diseases and neurologic lesions

where cognition can help evaluating the nonmotor neurological some of them known as cognitive symptoms are [1]: cerebral palsy, muscular dystrophy, multiple sclerosis, Parkinson's disease, traumatic brain injury, brain tumors, spinal cord injury, and others as shown in Fig. 1.14. Where:

- “*Cerebral palsy (CP)*” is a group of permanent disorders that appear in the childhood. “*CP*” is caused by abnormal development or damage to the parts of the brain that control movement, balance, posture and other areas as sensation, vision, hearing, swallowing, and speaking. The “*CP*” basic symptoms can be divided on “*motor*,” and “*nonmotor*”:
 - “*Typical motor symptoms*” vary among people that include poor coordination, stiff muscles, weak muscles, and tremors.
 - “*Frequently nonmotor neurological/cognitive symptoms*” are intellectual disorders, learning disorders, attention deficit hyperactivity disorder (*ADHD*), behavioral problems, visual/hearing impairments, speech, and language issues (*dysarthria*), sensory impairments/pain, seizures, epilepsy and others.
 “*CP*” can be managed, but there are no solutions for total cure. The current goal is reducing symptoms and limit brain damage, there is hope with *stem cell therapy* [54].
- “*Muscular dystrophy (MD)*” as a group of diseases that result in increasing weakening and breakdown of skeletal muscles over time. The disorders differ in

which muscles are primarily affected, degree of weakness, speed of worsening, and when symptoms begin to affect at the patient. Many people eventually become unable to walk or develop problems with other organs. There are several different types of “*muscular dystrophy*.” “*Muscle weakness*” is a hallmark of each type. But the symptoms can vary and start at different ages [55]. The “*MD* basic symptoms can be divided in motor and nonmotor”:

- “*Typical motor symptoms*” are: trouble walking, difficult raising the front of their foot (known as *foot droop*), frequently falling, curved spine (*scoliosis*), swallowing problems, with time they become weaker and weaker, losing the ability to sit, walk, and lift objects. Because the disease can also affect muscles in the heart and lungs, shortness of breath and abnormal heart rhythms can occur.
- “*Frequently nonmotor neurological/cognitive symptoms*” are mild intellectual impairment, learning disabilities, behavioral problems, daytime sleepiness, and others.

Actually, there is no cure for *MD*. Physical therapy, orthotics, and corrective surgery may help with some symptoms.

- “*Multiple sclerosis (MS)*” is a condition when the insulating covers of nerve cells in the brain and spinal cords are damaged, this is why is also known as “*demyelinating disease*.” This damage impairs the conduction of signals in the affected nerves, reducing the conduction ability causing deficiency in sensation,

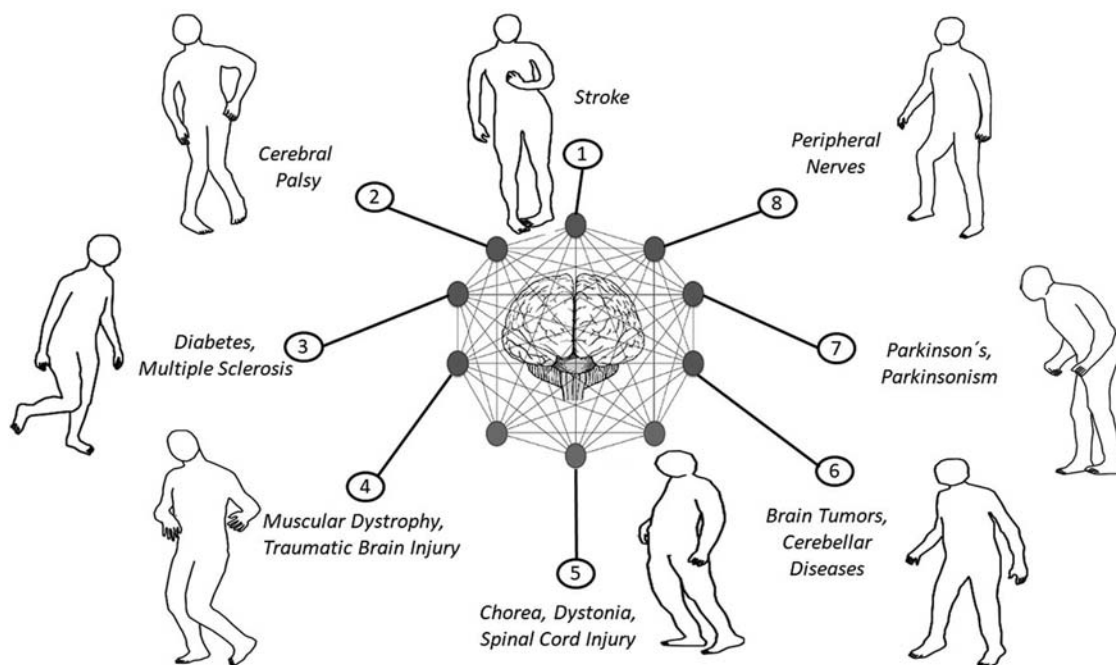


FIGURE 1.14 Some neurologic diseases with nonmotor cognitive symptoms.

movement, cognition, or other functions depending on which nerves are involved [56]. The “*MS basic symptoms can be motor and nonmotor*”:

- “*Typical motor symptoms*” are stiff muscles and spasms, sudden involuntary movements known as spasticity, trouble with walking, some degree of tremor, or uncontrollable shaking,
- “*Frequently nonmotor neurological/cognitive symptoms*” are:
 - “*Cognitive difficulties*” as memory loss, attention and concentration issues, difficulty processing information, trouble planning and prioritizing, and verbal fluency issues (word recall).
 - “*Vision problems*” as potential blurred vision, loss of normal color vision, blindness in one eye, a dark spot in the field of vision, double vision, and uncontrolled eye movements.
 - “*Fatigue*” from doing simple things, people with MS may also experience mental fatigue from depression.
 - “*Pain*” as burning, aching, and tingling “*pins and needles*” are also common around the body.

There is not yet a cure for “*MS*,” there are only many effective medications to help manage the disease.

- “*Parkinson’s disease (PD)*” is a progressive nervous system disorder that affect movement and present other symptoms, that can be different for everyone and the exact cause of this damage is still unknown [57]. The “*PD basic symptoms can be motor and nonmotor*”:
- “*Typical motor symptoms*” are tremors/shaking in the limbs, rigidity known as “*muscle stiffness*,” slowness of movements known as “*bradykinesia*,” and postural instability including impaired balance and/or difficulty standing or walking.
- “*Nonmotor neurological/cognitive symptoms*” are fatigue, digestive issues, sleep problems, cognitive changes including memory difficulties, slowed thinking, confusion, impaired visual-spatial skills such as getting lost in familiar locations, and dementia.

Parkinson’s disease cannot be cured, medications and surgeries might significantly improve their symptoms for some time.

- “*Traumatic brain injury (TBI)*” is a complex injury with a broad spectrum of symptoms. Since the brain defines who we are, the consequences of a brain injury can affect all parts of the life “*TBI*” patients, including personality. Injury specific areas of the brain will cause certain symptoms. For example, injury in the “*frontal lobes*” will cause loss of higher “*cognitive functions*,” such as loss of inhibition showing inappropriate social behavior. If the injury is in the

“*cerebellum*” then it will cause loss of coordination, balance, etc. The “*TBI basic symptoms can be motor and nonmotor*”:

- “*Typical motor symptoms*” are loss of consciousness, repeated vomiting or nausea, convulsions or seizures, weakness in fingers and toes, loss of coordination, and others.
 - “*Nonmotor neurological/cognitive symptoms*” are profound confusion, agitation/combativeness or other unusual behavior, slurred speech, coma and other disorders of consciousness
- No two brain injuries are alike, and the consequences of two similar injuries may be quite different. Beside the “*loss of consciousness (LOC)*,” commonly a person does not realize that a brain injury has occurred, symptoms may appear right away, or may not be present for days or weeks after the injury. Recovery is a functional aspect, based on a mechanism that remains uncertain [58].
- “*Brain tumors*” also known as “*intracranial neoplasm*” occur when abnormal cells form within the brain. Two types of “*brain tumors*” exist: “*malignant (cancerous)*” and “*benign tumors*.” The symptoms vary depending on the part of the brain affected; the “*general signs and symptoms can basic be divided in two motor and nonmotor*” [59]:
 - “*Typical motor symptoms*” are present, these could be headaches, seizures or convulsions, vomiting, and more specific problems include difficulty in walking and balance, fatigue, sleep problems, repetitive movements, and others.
 - “*Nonmotor neurological/cognitive symptoms*” are changes in sensation, vision, smell, hearing, personality, memory changes, changes in emotional state, and more.

Brain tumors treatments are surgery to remove the brain tumor, radiation therapy applying a high-powered ray to damage cancer cells and stop them from growing, and chemotherapy using of drugs to kill cancer cells.

- “*Spinal cord injury (SCI)*” occurs when signals between your brain and body are disrupted. These cause problems such as weakness and paralysis. The spinal cord is a bundle of nerves carrying signals back and forth between the body and the brain. The most common injuries are when pieces of vertebrae tear cord tissues or press down on the nerve parts that carry signals, others occur with fractures or dislocation of vertebrae [60]. The lowest normal part of your “*spinal cord*” is referred to as the neurological level of your injury. The severity of the injury is often called “*the completeness*” and is classified as either of the following: “*Complete*” if all sensory feeling and all abilities to control movement as a

motor function are lost below the spinal cord injury, and “*Incomplete*.” If “*SCI*” presents some motor or sensory function below the affected area. The “*general signs and symptoms can basic be divided in two motors and nonmotor*”:

- “*Typical motor symptoms*” are: loss of movements identified as “*tetraplegia or quadriplegia*” if a paralysis is present in trunk, both arms, both legs and pelvic organs are all affected, and “*Paraplegia*” if paralysis is present part of the trunk, legs and pelvic organs.
- “*Nonmotor neurological/cognitive symptoms*” are loss of sensations in hands, fingers, feet or toes, anxiety, depression, pain, and others.

There is no treatment to reverse damage to the SCI, but there is hope on stem cell research.

- Others neurologic diseases and lesions as: *stroke, diabetes mellitus, seizures disorders, Guillain-Barré syndrome, Post-Polio syndrome, and many others that present motor symptoms and nonmotor neurological or cognitive symptoms.*

“*Neuroscience*” and “*Cognitive Science*” has help in the success of today’s “*AI*” algorithms as “*DL*” in building a system that mirrored the simulations of the human brain known today as “*Artificial Neural Networks (ANNs)*” and “*AI*” is going to help more applying the *Architecture framework of AI-Cognitive Computing Agents System (AI-CCAS)* to evaluate, detect, analyze, classify and forecast nonmotor neurological/cognitive symptoms present in many neurological diseases. In [Chapter 7](#), *Cognitive Learning and Reasoning Models Applied to Biomedical Engineering*, see “*Cognitive Learning and its relationship with neuroscience of reasoning proposed as Cognitive Learning-Reasoning (CL&R) using Cognitive Computing (CC).*”

1.7.1 The near future of neuroscience, cognitive science, and AI-ML-DL-CC

There is a boom on “*DL application using mathematical algorithms for different types of ANNs*” because they are based on the design of individual neurons that are connected across multiple neurons creating paths for processing complex concepts. But “*ANNs*” are still a simplified model of the biological human brain with the only objective of detect and classify patterns in images and text, it acquires knowledge through learning, and stores this knowledge by adjusting the weights within the network making this process too slow when multiple layers led to increase training time and accurate. Then, it is needed to run “*ANNs algorithms*” on special powerful processor as the “*Nvidia GPUs*” for fast processing and even in some ANN algorithms are necessary to use high-performance

parallel computing on GPUs as “*NVIDIA CUDA.*” This evolution seems to point in the direction of revolutionaries’ new software technologies based in “*new AI learning methods*” and “*new auto-configurable hardware*” needed to develop “*living biohybrid systems.*” Some examples are:

- *New AI learning software:*
 - “*AI-ML-DL-CC*” can be used to better simulating of brain behaviors, so that “*cognitive neuroscientists*” can test using “*Cognitive Computing Agents Systems*” as the one shown in [Fig. 1.13](#), where their “*AI models*” can produce outputs which agree with the responses given by biological neural networks to evaluate patients with neurological disorders.
 - Associative Structures method can be used as AI learning model. Where “*Associative Structures*” are created by the human brain, when many related views that a person learns to associate one thing with another due to a previous experience with it. With the results, “*cognitive psychologists*” can explore ideas of association of ideas that can be explained in terms of an associative structure tests evaluated by “*Architecture framework of AI-Cognitive Computing Agents System (AI-CCAS).*”
 - Connectionism technique can be used studied as a base for a new AI learning model. Where “*Connectionism*” is the theory that all mental processes can be described as the operation of inherited or acquired bonds between stimulus and response evaluated by “*Architecture framework of AI-Cognitive Computing Agents System (AI-CCAS).*”
 - Transfer learning also known as Inductive learning can be further improved applied on AI-ML-DL-CC, where “*Transfer Learning*” is when a system focuses on storing knowledge gained while solving one problem and apply it to a different related problem. Some actual examples are the transfer learning with image data that can be downloaded and incorporated directly into new models that expect data input as [\[61\]](#): Oxford VGG Model [\[62\]](#), Google Inception Model [\[63\]](#), Microsoft ResNet Model [\[64\]](#), Caffe Model Zoo [\[65\]](#), the development of progressive Neural Network for transfer learning [\[66\]](#), etc.
 - And many other new learning models techniques that are continuously evolving.
- *New auto-configurable architecture hardware:*
 - “*Neural network computer*” has a computer architecture in which a number of “*Neural Network Processors (NNP)*” are interconnected in a manner suggestive of the connections between neurons in a “*human brain*” with variable software connections as serial and/or parallel. Where the traditional RAM

digital memory is replaced by “*threshold memory based on ANN*” as the “*Neural Turing Machine*” or “*Differentiable Neural Computers*” or “*Liquid State Machine based on Spiking Neural Network*” or other. “*Neural network computer*” is able to learn by a process of trial and error, and many other reasoning methods as explained in Chapter 7, “*Cognitive Learning and Reasoning Models Applied to Biomedical Engineering*,” of this book.

- *Biomolecular computers* that use systems of biologically derived molecules such as “*DNA and proteins*” to perform computational calculations involving storing, retrieving, and processing data. The development of biocomputers has been made possible by the expanding new science of “nanobiotechnology, as the *DNA computers*,” that has “*biomolecular components*” rather than standard artificial hardware using silicon chips in computer technology. In place of traditional binary code, “*DNA computing*” utilizes the four-character genetic alphabet, which consists of: A—Adenine, G—Guanine, C—Cytosine, and T—Thymine) [67],
- “*Living biohybrid systems*” defined as a system that have configurable hardware, that can connect and exchange information between biological systems, like neurons in the brain, and human-made electronic to help in “*AI models*” to develop new applications needed for *Cognitive Science and Neuroscience* [68].
- And many others at the development stage as the Quantum computer. Where Quantum computing is the amalgam of Physics, Mathematics, and Quantum Mechanics that exploits the collective properties of quantum states, such as superposition, interference, and entanglement, to perform computation. The devices that perform quantum computations are known as quantum computers [69].

Cognitive science and neuroscience have evolved over the years, and recently they have started to intersect. This is very useful for the development of new *Biomedical Engineering* solutions applying tools based on *Artificial Intelligence—Machine Learning—Deep Learning—Cognitive Computing (AI-ML-DL-CC)*. Their evolution seems to point in the direction of revolutionary new software technologies based on “*new AI learning methods*” and “*new auto-configurable hardware*,” such as “*neural network computers*,” “*biomolecular computers*,” “*living biohybrid systems*,” and others that are currently in research and development. The concept of “*Cognitive Learning and its relationship with the neuroscience of reasoning—proposed as Cognitive Learning-Reasoning (CL&R)*—” will be easier and faster to implement.

References

- [1] J. Garza-Ulloa, Paperback ISBN: 9780128125946, eBook ISBN: 9780128125953 June Applied Biomechatronics Using Mathematical Models, first ed., Academic Press Elsevier, 2018.
- [2] J. Garza-Ulloa, Chapter 1 - Introduction to biomechatronics/biomedical engineering, in: J. Garza-Ulloa (Ed.), Applied Biomechatronics Using Mathematical Models, Academic Press, 2018, pp. 1–51. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00001-9>. <http://www.sciencedirect.com/science/article/pii/B9780128125946000019>. ISBN 9780128125946.
- [3] IEEE. Available from: <https://cis.ieee.org/about/what-is-ci> (accessed 15.07.19).
- [4] Available from: <https://www.healthcatalyst.com/data-mining-in-healthcare> (accessed 15.07.19).
- [5] S. Costantini, Meta-reasoning: A Survey (2001). doi: 10.1007/3-540-45632-5_11.
- [6] D. Berrar, M. Granzow, W. Dubitzky, Introduction to genomic and proteomic data analysis, in: W. Dubitzky, M. Granzow, D. Berrar (Eds.), Fundamentals of Data Mining in Genomics and Proteomics, Springer, Boston, MA, 2007.
- [7] O. Bodenreider, J.A. Mitchell, A.T. McCray, Biomedical ontologies, Pac. Symp. Biocomput. (2005) 76–78. PubMed PMID: 15759615; PubMed Central PMCID: PMC4300097.
- [8] J.W. Funder, Medicine as a profession, Clin. Med. (Lond.) 10 (3) (2010) 246–247.
- [9] S.A. Wartman, C.D. Combs, Medical education must move from the information age to the age of artificial intelligence, Acad. Med. 93 (8) (2018) 1107–1109.
- [10] S.A. Wartman, C.D. Combs, Reimagining medical education in the age of AI, AMA J. Ethics 21 (2) (2019) E146–E152. Available from: <https://doi.org/10.31478/202109a>.
- [11] J.H. Chen, S.M. Asch, Machine learning and prediction in medicine—beyond the peak of inflated expectations, N. Engl. J. Med. 376 (26) (2017) 2507–2509.
- [12] C. Smith, B. McGuire, T. Huang, et al., The History of Artificial Intelligence, University of Washington, 2006.
- [13] A. Turing, Computing Machinery and Intelligence, 1950. Available from: <http://www.abelard.org/turpap/turpap.htm>.
- [14] Available from: <https://www.britannica.com/topic/laws-of-thought> (accessed 15.06.19).
- [15] Available from: <https://deepmind.com/research/alphago/> (accessed 15.06.19).
- [16] Available from: <https://theconversation.com/understanding-the-four-types-of-ai-from-reactive-robots-to-self-aware-beings-67616> (accessed 15.06.19).
- [17] Available from: <https://www.govtech.com/computing/Understanding-the-Four-Types-of-Artificial-Intelligence.html> June 2019.
- [18] M. Hermann, T. Pentek, B. Otto, Design Principles for Industrie 4.0 Scenarios, in: 2016 49th Hawaii International Conference on System Sciences (HICSS), 2016, pp. 3928–3937. Available from: <https://doi.org/10.1109/HICSS.2016.488>.
- [19] B. Marr, Why everyone must get ready for the 4th industrial revolution, Forbes, 2018.
- [20] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006. ISBN 978-0-387-31073-2.
- [21] M. Claesen, B. De Moor, Hyperparameter search in machine learning. arXiv:1502.02127, 2015.

- [22] J. Garza-Ulloa, Chapter 2 - Introduction to human neuromusculoskeletal systems, *Applied Biomechanics Using Mathematical Models*, Academic Press, 2018, pp. 53–118. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00002-0>. <http://www.sciencedirect.com/science/article/pii/B9780128125946000020>. ISBN 9780128125946.
- [23] S. Haykin, *Neural Networks: A Comprehensive Foundation*, second ed., Prentice-Hall, Englewood Cliffs, NJ, 1999.
- [24] R. Rojas, *Neural Networks*, Springer-Verlag, Berlin, 1996. ISBN 978-3-642-61068-4.
- [25] Available from: <https://web.archive.org/web/20101218121158/http://herselfsai.com/2007/03/probabilistic-neural-networks.html> (accessed 26.07.19).
- [26] Available from: <http://www.psi.toronto.edu/~vincent/research/presentations/PNN.pdf> (accessed 26.07.19).
- [27] Available from: https://en.wikipedia.org/wiki/Probabilistic_neural_network (accessed 26.07.19).
- [28] N.-D. Hoang, D.T. Bui, Chapter 18 - slope stability evaluation using radial basis function neural network, least squares support vector machines, and extreme learning machine, in: P. Samui, S. Sekhar, V. E. Balas (Eds.), *Handbook of Neural Computation*, Academic Press, 2017, pp. 333–344. ISBN 9780128113189. Available from: <https://doi.org/10.1016/B978-0-12-811318-9.00018-1>. Available from: <http://www.sciencedirect.com/science/article/pii/B9780128113189000181>.
- [29] Available from: <https://towardsdatascience.com/ml-for-ts-3-extreme-learning-machines-3fcf5991e390> (accessed 25.07.19).
- [30] W. Badr, Auto-Encoder: what is it? And what is it used for?. Available from: <https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726> (accessed 24.07.19).
- [31] D. Monn, Denoising Auto-Encoders explained. Available from: <https://towardsdatascience.com/denoising-autoencoders-explained-dbb82467fc2> (accessed 24.07.19).
- [32] G. Liu, H. Bao, B. Han, A stacked Auto-Encoder-based deep neural network for achieving gearbox fault diagnosis [online], *Hindawi*, 2018. Available from <https://www.hindawi.com/journals/mpe/2018/5105709/> (accessed 28.11.18).
- [33] B. Jefkine, Introduction. *Convolutional Neural Networks*, September 2016. Available from: <https://jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/> (accessed 27.07.19).
- [34] S. Saha, A comprehensive guide to convolutional neural networks—the ELI5 way, December 2018. Available from: <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53> (accessed 27.07.19).
- [35] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, et al., Recent advances in convolutional neural networks, *Pattern Recognit.* 77 (2018) 354–377 ISSN 0031-3203. Available from: <https://doi.org/10.1016/j.patcog.2017.10.013>. <http://www.sciencedirect.com/science/article/pii/S0031320317304120>.
- [36] D.M. Zeiler, D. Krishnan, et al., Deconvolutional networks. Available from: <https://cs.nyu.edu/~fergus/drafts/utexas2.pdf> (accessed 25.07.19).
- [37] T.D. Kulkarni, W. Whitney, et al., Deep convolutional inverse graphics network. Available from: <http://willwhitney.com/dc-ign/www/> (accessed 25.07.19).
- [38] J. Rocca, Understanding generative adversarial networks (GANs). Available from: <https://towardsdatascience.com/understanding-generative-adversarial-networks-gans-cd6e4651a29> (accessed 25.07.19).
- [39] Available from: https://en.wikipedia.org/wiki/Long_short-term_memory (accessed 24.07.19).
- [40] A. Ghosh, Recurrent neural network and long short term memory, October 2018. Available from: <https://medium.com/datadriveninvestor/recurrent-neural-networks-and-long-short-term-memory-5d17bdbdfc00> (accessed 24.07.19).
- [41] C.J. Spoeer, P. McClure, N. Kriegeskorte, Recurrent convolutional neural networks: a better model of biological object recognition. *Front. Psychol.*, September 12, 2017. Available from: <https://doi.org/10.3389/fpsyg.2017.01551> (accessed 12.08.19).
- [42] H. Hananel, M. Larry, Manevit, Topological constraints and robustness in liquid state machines, *Expert. Syst. Appl.* 39 (2) (2012) 1597–1606. Available from: <https://doi.org/10.1016/j.eswa.2011.06.052>.
- [43] Available from: <https://www.osti.gov/servlets/purl/1405258> (accessed 25.07.19).
- [44] H. Jaeger, H. Haas, *Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication*, *Science* 304 (5667) (2004) 78–80.
- [45] T. Kohonen, Self-organized formation of topologically correct feature maps, *Biol. Cybern.* 43 (1) (1982) 59–69.
- [46] A. Graves, G. Wayne, I. Danihelka. *Neural Turing machines*. Available from: <http://www.robots.ox.ac.uk/~tvj/publications/talks/NeuralTuringMachines.pdf> (accessed 26.07.19).
- [47] Available from: <https://deepmind.com/blog/differentiable-neural-computers/> (accessed 27.07.19).
- [48] Available from: <http://www.asimovinstitute.org/neural-network-zoo/> (accessed 27.07.19).
- [49] S. Sabour, N. Frosst, G.E. Hinton (October 26, 2017). Dynamic routing between capsules. arXIV.
- [50] Available from: <http://www.asimovinstitute.org/neural-network-zoo/> (accessed 30.07.19).
- [51] H. Sankesara, Hierarchical attention networks, August 2018. Available from: <https://medium.com/analytics-vidhya/hierarchical-attention-networks-d220318cf87e> (accessed 30.07.19).
- [52] A. Chandio, D.K. Chaturvedi, CIT: Integrated cognitive computing and cognitive agent technologies based cognitive architecture for human-like functionality in artificial systems, *Biol. Inspired Cogn. Archit.* 26 (2018) 55–79 ISSN 2212-683X. Available from: <https://doi.org/10.1016/j.bic.2018.07.020>. <http://www.sciencedirect.com/science/article/pii/S2212683X18300033>.
- [53] S. Adams, I. Arel, J. Bach, R. Coop, R. Furlan, B. Goertzel, et al., Mapping the landscape of human-level artificial general intelligence, *AI Mag.* 33 (1) (2012) 25–42.
- [54] Available from: <https://www.cerebralpalsyguide.com/treatment/cure/> (accessed 06.08.19).
- [55] Available from: <http://mda.org/> (accessed 14.08.19).
- [56] Available from: <https://www.mayoclinic.org/diseases-conditions/multiple-sclerosis/multimedia/multiple-sclerosis-diagnosis/vid-20135054> (accessed 14.08.19).
- [57] J. Garza-Ulloa, Update on Parkinson’s disease, *Am. J. Biomed. Sci. Res.* 2 (6) (2019). Available from: <https://doi.org/10.34297/AJBSR.2019.02.000614>. AJBSR.MS.ID.000614.
- [58] Available from: <https://www.mayoclinic.org/diseases-conditions/traumatic-brain-injury/symptoms-causes/syc-20378557/> (accessed 06.08.19).
- [59] Available from: <https://www.cancer.net/cancer-types/brain-tumor/symptoms-and-signs> (accessed 06.08.19).

- [60] Available from: <https://medlineplus.gov/spinalcordinjuries.html> (accessed 06.08.19).
- [61] Available from: <https://machinelearningmastery.com/transfer-learning-for-deep-learning/> (accessed 06.08.19).
- [62] Available from: http://www.robots.ox.ac.uk/~vgg/research/very_deep/ (accessed 06.08.19).
- [63] Available from: <https://github.com/tensorflow/models/tree/master/research/inception> (accessed 06.08.19).
- [64] Available from: <https://github.com/KaimingHe/deep-residual-networks> (accessed 06.08.19).
- [65] Available from: <https://github.com/BVLC/caffe/wiki/Model-Zoo> (accessed 06.06.19).
- [66] J. Gideon, S. Khorram, Z. Aldeneh, D. Dimitriadis, E.M. Provost, Progressive neural networks for transfer learning in emotion recognition, INTERSPEECH 2017, Stockholm, Sweden, August 20–24, 2017.
- [67] Y. Benenson, Biomolecular computing systems: principles, progress and potential. Nat. Rev. Genet. (2012) [online]. Nature Publishing Group, a division of Macmillan Publishers Limited. All Rights Reserved. <https://doi.org/10.1038/nrg3197>.
- [68] Available from: <https://www.sciencedaily.com/releases/2016/11/161115111647.htm> (accessed 15.08.19).
- [69] Available from: [Quantum Computing](#), 1, 2021.

This page intentionally left blank

Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive relation to help in the solution of Artificial Intelligence Biomedical Engineering problems

2.1 Introduction

The analysis for applying multidisciplinary engineering principles of medicine and biology for health purposes for body injuries, illness, and neurological disorders can be separated by the typical symptoms of these disorders. They are generally divided into: “*motor symptoms related to movement disorders*” and “*nonmotor symptoms related to cognition and no related to movement disorders*,” where:

- “*Motor symptoms*” are the human body movement disorders of the patients, that can be subclassified as: “*motor*” that are directly related to movement, and “*secondary motor*” when they are consequences of movement disorders [1]:
 - “*Motor**”: tremors (shaking in the limbs), rigidity (muscle stiffness), bradykinesia (slowness of movements), postural instability (impaired balance or difficulty standing or walking), and others.
 - “*Secondary motor**”: hypomimia (loss of facial expressions), freezing of gait or shuffling gait, unwanted accelerations in body movements and speech, dystonia (involuntary muscle contractions), speech difficulty, and others.

Note*: The “*motor*” and “*secondary motor*” symptoms were studied in my book [2]: “*Applied Biomechanics Using Mathematical Models*,” which provides an appropriate methodology to detect and measure diseases and injuries relating to human kinematics and kinetics. It features mathematical models that, when applied to engineering principles and techniques in the medical field, can be

used in assistive devices that work with bodily signals. It is shown Fig. 2.1A.

- “*Nonmotor symptoms*” refer to all the other human disorders that are present in patients with body injuries or neurologic diseases, that are not related to movement disorders. These “*nonmotor symptoms*” can be subclassified as: “*cognitive changes (affecting the way the think, behavior, and mental status)*” and “*specific nonmotor symptoms changes*,” where:
 - “*Cognitive changes*” can be detected as memory difficulties, slow thinking, confusion, misbeliefs, dementia, mood alteration, psychotic problems, and many other alterations on human cognition.
 - “*Specific nonmotor symptoms changes*” are fatigue, digestive issues, sleep problems, orthostatic hypotension, increased sweating, increased drooling, pain, hyposmia (reduce sense of smell), melanoma, and many others.

The “*nonmotor symptoms*” and “*others physical and mental changes*” are studied as “*cognitive changes*” analysis in this book under cognitive computing. As well as general body injuries, illness and disorders are the focus of this book: “*Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models*,” as shown in Fig. 2.1B).

Note: It is important to mention that is not necessary to read the books in sequence, but I will make some reference to the other book when is necessary to avoid the

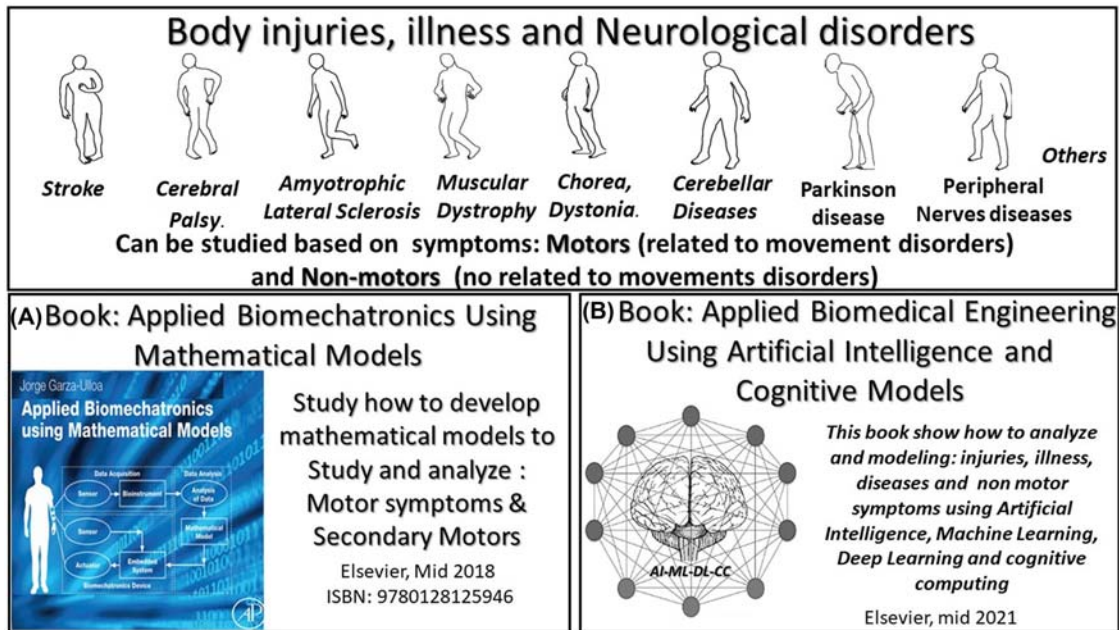


FIGURE 2.1 Injuries and neurologic diseases are studied in two books: (A) motor symptoms in “*Applied Biomechanics Using Mathematical Models*”; and (B) nonmotor cognitive systems in “*Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models*.”

duplication of explanations of concepts, themes, and figures

The main goal of this book is based on the following hypothesis:

“If we understand how the developed brain achieves cognitive abilities, and how the functional interactions between the distributed brain regions work to produce the sophisticated human cognitive systems, we could develop someday more realistic models based on Artificial Intelligence (AI) algorithms to detect cognition abnormalities in the human brain compared with normal cognition, and find ways to understand and evaluate injured neurologic human systems with the goal to help find better solutions to stop the progression, and improve prognosis of many neurologic diseases that still don’t have cures.”

2.2 Brain, spinal cord, and nerves

The “*motor symptoms detected as movement disorders*” were studied in my previous book: “*Applied Biomechanics Using Mathematical Models*,” in “*Chapter 2, Introduction to Human Neuromusculoskeletal Systems*” [3]. It can be summarized as “*the study of different methods to obtain mathematical models and apply them to evaluate human motion, taking into account the effect of how the human processes information and acts when it needs to make body movements, based on the interaction between*

different organs systems,” such as the “*nervous system*” and “*musculoskeletal system.*”

- “*Nervous system*” is integrated by the “*central nervous system (CNS)*” and “*peripheral nervous system (PNS)*,” where:
 - “*CNS*” consists of the “*brain and the spinal cord.*”
 - “*PNS*” consists of “*sensory nerves and the sense organs.*”
- “*Musculoskeletal system*” integrated by “*skeletal system*” and “*muscular system*”:
 - “*Skeletal system*” is composed of bones and joints in the body, including bones, cartilage, and ligaments.
 - “*Muscular system*” is composed of muscles of three kinds: skeletal, smooth, and cardiac.

In general, the “*nervous and musculoskeletal systems**” control how the skeletal system and the muscular system work together as the framework for the body, and provide orders for the movements that are controlled by the nervous systems through the “*CNS*” and “*PNS.*”

Note*: See more information in the book: “*Applied Biomechanics Using Mathematical Models*,” especially in: “*Chapter 5, Methods to Develop Mathematical Models: Traditional Statistical Analysis*” [4] and “*Chapter 6, Application of Mathematical Models in Biomechanics: Artificial Intelligence and Time-Frequency Analysis*” [5].

The “*nonmotor symptoms*” specially the “*cognitive changes*” that affect the way of thinking, behavior, and

mental status are studied in this chapter. We will introduce “*Jean Piaget’s theory of cognitive development*,” which suggests, along with the discussions of many other researchers [6–8], that children move through basically four different stages of mental development [9]: “*sensorimotor stage*,” “*preoperational stage*,” “*concrete operational stage*,” and “*formal operational stage*.”

- “*Sensorimotor stage*” from birth to 2 years. In this first stage for cognitive development they “*acquire knowledge through sensory experiences and the basic manipulation of objects*” using basic human senses, motor responses, and basic reflexes.
- “*Preoperational stage*” from age 2 to 7. In this second stage they develop “*language, and concentrate on the world around them, applying basic logic principles*,” begin taking the point of view of other people, and gain a basic understanding of consistency with their ideas.
- “*Concrete operational stage*” from age 7 to 11. Here they begin to using “*inductive logic or reasoning*” based on specific information available; their thinking becomes more logical and organized in concrete events.
- “*Formal operational stage*” from age 12 and up. In the final stage they gain the “*ability to thinking about abstract ideas and situations*,” which is the key hallmark of the formal operational stage of “*cognitive development*.”

These “*uniquely human abilities are made possible by a protracted trajectory of brain development and learning over the first two decades of life [10] and continue with the formation of neural pathways and cognition levels during our life*” [9].

In this book we concentrate on the general way in injuries, diseases and specific on “*nonmotor cognitive symptoms*” based and neurological diseases disorders that generate “*cognitive decline*,” as illustrated in upper of Fig. 2.1, to study how and where the “*cognitive functional brain networks*” execute orders for: “*language*,” “*reasoning and general cognitive control*” [11] and how to evaluate them based on methods of “*mathematical algorithms from Artificial Intelligence (AI), Machine Learning (ML), Deep Learning (DL), and Cognitive Computing (CC)*.”

2.3 Neurons and neural pathways in cognition

Different kinds of “*neurons*” and their “*neural pathways*” are used for “*language*,” “*reasoning*,” and “*general cognitive control*” in the human being. When each “*neuron eventually comes a neural pathway, these pathways play*

a very large role in the way we perceive and act in different situations.” These “*human behaviors and responses are known as cognition*”; they are the result of how the preexisting pathways process the information, and their undeniable ability to reshape themselves under different circumstances. These actions direct the importance of studying the relation between “*Neurons and cognition*,” “*Neural pathways and cognition*,” and one special group of “*Neurotransmitter disorders*” identified as “*Dopamine pathways and cognition*.”

2.3.1 Neurons and cognition

The “*neuron*” is a specialized “*cell*” known as the basic unit of the “*nervous system*.” A typical neuron consists of a “*dendrite*,” a cell body known as the “*soma*,” “*axon hillock*,” “*axon*,” and “*axon terminals*,” as shown at the top of Fig. 1.6A. A “*type I synapse provides an excitatory connection*” between one “*axon terminal*” of a neuron to the “*dendrite*” of another neuron. The other “*type II inhibitory connection of synapses*” is typically located on a cell body. The cell that sends out information is called a “*pre-synaptic neuron*,” and the cell that receives information is known as a “*postsynaptic neuron*” [3], as shown in Fig. 2.2A. It is important to mention that there are more specialized types and subtypes of “*neurons*” that form an extensive “*neuron taxonomy*”; a good resource is the digital database that can be found on the website “*NeuroMorpho.org*” [12]. One practical reason is that the differences between them could explain why certain diseases only harm a certain population of “*neurons*” [13]. The “*neuron’s function*” is based on two kinds of activities: “*electrical*” and “*chemical*.”

- “*Electrical activity is used to transmit signals within neurons*.” “*Neurons*” employ electrical signals to relay information from one part of the neuron to another. “*Within a single neuron, information is conducted via electrical signaling*.” When a “*neuron*” is stimulated, an electrical impulse, called the “*action potential*,” moves along the “*neuron axon*,” which is a long threadlike part of a nerve. The “*action potential*” enables signals to travel very rapidly along the “*neuron*.”
- “*Chemical activity is used to transmit signals between neurons through a small gap that separates neurons, known as synapse*,” as shown in Fig. 2.2A. These trigger the release of “*neurotransmitters*” which carry the impulse across the “*synapse to the next neuron*.” Once a nerve impulse has triggered the release of “*neurotransmitters*,” these “*chemical messengers*” cross the tiny “*synaptic gap*” and are taken up by specialized receptors on the surface of the next cell, as indicated at the bottom of Fig. 2.2A. This process converts the

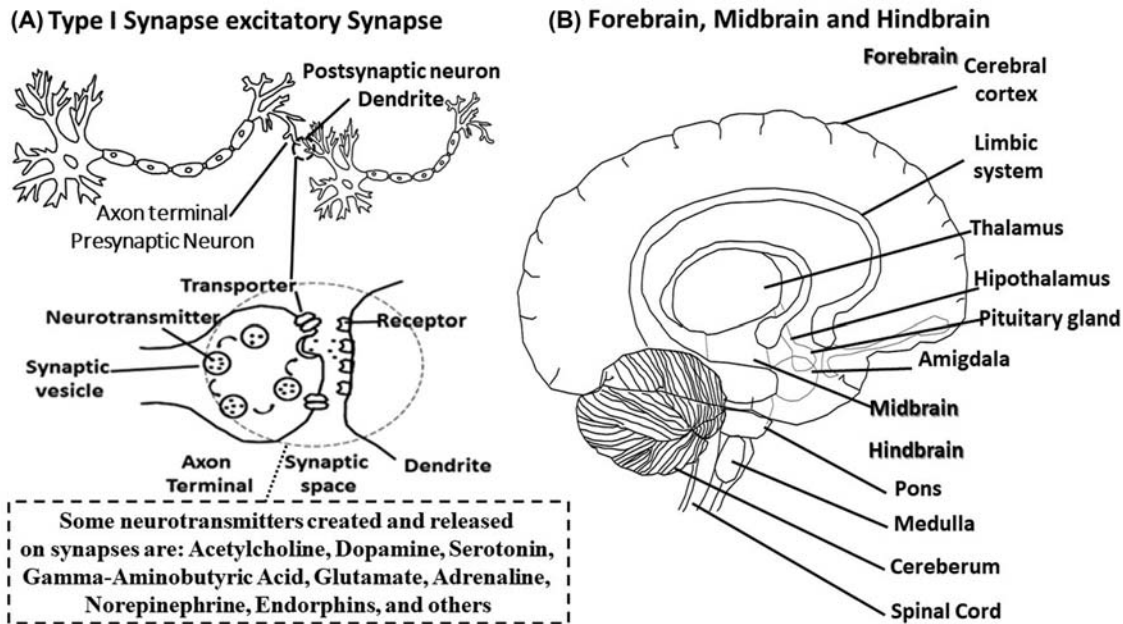


FIGURE 2.2 (A) Two neurons joined by type I excitatory synapse, and some major neurotransmitters released during the process. (B) The three major regions and some of their components in the brain: forebrain, midbrain, and hindbrain.

chemical signal back into an electrical signal. If the signal is strong enough, it will be propagated down to the next neuron by an “*action potential*” until once again it reaches a “*synapse*” and the process is repeated once more.

“*Synapse*”s are located throughout the “*brain*” and “*nervous system*” and refer to the junction between two neurons. They behave as a sort of “*relay station*” where a message in the form of a chemical “*neurotransmitter*” is passed from one “*neuron or nerve fiber*” to the next, or between the “*neuron and the muscle or gland*” the message is aimed at. On average, each “*neuron has around 1000 synapses and depending on its type can have from just one to more than 1000 synapses.*”

“*Neurotransmitters*” are chemicals that transmit signals from a “*neuron*” to a “*target cell*” across a “*synapse*.” There are different types of these small molecules manufactured in different kinds of “*axon terminals*.” The major classes of them include “*amino acids*,” “*peptides*,” and “*monoamines*.” Some important “*neurotransmitters*” are shown in Fig. 2.2A: “*acetylcholine*,” “*dopamine*,” “*serotonin*,” “*gamma-aminobutyric acid*,” “*glutamate*,” “*epinephrine or adrenaline and norepinephrine*,” “*endorphins*,” and others. The specific function of each “*neurotransmitter*” is as follows:

- “*Acetylcholine (Ach)*” is used by the “*CNS*” and “*PNS*” to cause muscle contraction, and many “*neurons in the brain to regulate memory*.” In most instances, “*Ach*” has an “*excitatory function*,” and it is

one of many “*neurotransmitters*” in the “*autonomic nervous system*,” and the only “*neurotransmitter*” used in the motor division of the “*somatic nervous system*.”

- “*Dopamine (DA)*” is produced in few areas of the brain, including the “*substantia nigra*” and the “*ventral tegmental area*,” which is a group of “*neurons*” located close to the midline on the floor of the “*midbrain*” or “*mesencephalon*,” as indicated in Fig. 2.2B. “*DA*” is also a “*neurohormone*” released by the “*hypothalamus*,” and it has important roles in “*behavior and cognition, voluntary movement, motivation, punishment and reward, sleep, mood, attention, working memory, and learning.*”
- “*Serotonin (5-HT)*” is a monoamine “*neurotransmitter*,” usually found in the “*gastrointestinal tract*,” “*platelets*,” and the “*CNS*.” This chemical is also known as the “*happiness hormone*,” because it arouses feelings of pleasure and well-being. “*Low levels of serotonin are associated with increased carbohydrate cravings, depression or other mood symptoms, sensory perceptions, sleep deprivation, and hypersensitivity to pain.*”
- “*Gamma-aminobutyric acid (GABA)*” is the major inhibitory “*neurotransmitter*” in the brain. It is important in producing sleep, reducing anxiety, and forming memory. “*The primary role of GABA is to slow down neuron activity.*”
- “*Glutamate (Glu)*” is the most abundant excitatory neurotransmitter in the vertebrate nervous system. “*Glu is also the major excitatory transmitter in the*

brain, and the major mediator of excitatory signals in the mammalian central nervous system. It is involved in most aspects of normal brain function, including cognition, memory, and learning.”

- “Epinephrine or adrenaline (Epi) and norepinephrine (NE),” these are separate but related hormones secreted by the medulla of the “adrenal glands.” These chemicals are also produced at the ends of sympathetic nerve fibers, where “they serve as chemical mediators for conveying the nerve impulses to effector organs. They are responsible for concentration, attention, mood, and both physical and mental arousal.”
- “Endorphins” are produced by the “pituitary gland” and the “hypothalamus” in vertebrates during exercise, excitement, pain, consumption of spicy food, love, and orgasm. “Endorphins contribute to the feeling of well-being and act similarly to opiates. They are also known to reduce pain and anxiety.”

“We can deduce that any malfunction of synapses affects the creation and transition of neurotransmitters’ effects, as it can affect the order sent by the brain leading to many cognitive alterations. They are reflected as multiple symptoms in neurologic disorders” [14].

2.3.2 Neural pathway and cognition

A “neural pathway” is a bundle of “axons” that connects two or more different neurons, facilitating communication between them. These “cells make an up a massive network of specialized cells that transmit messages from one part of the body to another, creating complex interconnected circuits, that can function independently with parallel processing capability.” These “neural pathways” can be of three types, “motor pathways,” “sensory pathways” and “inter-neurons that connect neurons together”. Where:

- “Motor pathways” are “descending efferent nerve pathways,” and they travel from the “CNS to an effector organ, muscle, or gland.”
- “Sensory pathways” are “ascending afferent nerve paths,” and travel from the “sensory receptor in the skin, muscles, and tendons to the CNS.”

A “neural pathway” connects one part of the “nervous system” to another using bundles of “axons” called “tracts.” The “tracts” are named according to their origin in the first half of the term and its termination in the last half term, for example, the “spinothalamic tract” begins in the spinal cord and ends in the thalamus, the “corticospinal tract” begins in the cerebral cortex and finishes in the spinal cord, etc. The majors “neural pathways” are:

“arcuate fasciculus,” “cerebral peduncle,” “corpus callosum,” “pyramidal tracts,” “medial forebrain bundle,” “dorsal column-medial lemniscus pathway,” and “retino-hypothalamic tract” [15].

- “Arcuate fasciculus” is the neural pathway connecting the posterior part of the “tempoparietal junction” with the “frontal cortex” in the brain. The “arcuate fasciculus” helps in cognition by connecting areas of the brain involved in the generation and understanding of language; damage of this pathway can cause a form of “aphasia,” where auditory comprehension and speech articulation are preserved, but people find it difficult to repeat heard speech.
- “Cerebral peduncle.” The three common areas involved are: “brain cortex,” “spinal cord,” and “cerebellum.” “Cerebral peduncle” helps in cognition by assisting in the refined motor movements, including learning new motor skills and converting proprioceptive information into balance and posture maintenance.
- “Corpus callosum” connects the left and right cerebral hemispheres and facilitates interhemispheric communication. It is the largest white matter structure in the brain. The “corpus callosum helps in cognition by transferring motor, sensory, and cognitive information between the brain hemispheres.”
- “Pyramidal tracts” contain exclusively “motor axons.” They actually consist of two separate tracts in the spinal cord: the “lateral corticospinal tract” and the “medial corticospinal tract.” “Pyramidal tracts lead to the understanding of why in the most part, one side of the body is controlled by the opposite side of the brain.”
- “Medial forebrain bundle (MFB)” is a neural pathway containing fibers from the “basal olfactory regions,” “peri amygdaloid region,” and the “septal nuclei,” as well as fibers from “brainstem regions,” including the “ventral tegmental area.” The “MFB” is one of the two major pathways connecting the “limbic forebrain,” “midbrain,” and “hindbrain,” as indicated in Fig. 2.2B [16]. “MFB is a part of the reward system, involved in the integration of reward and pleasure [17]. Electrical stimulation of the medial forebrain bundle is believed to cause sensations of pleasure.”
- “Dorsal column-medial lemniscus pathway (DCML)” is also known as the “posterior column-medial lemniscus pathway (PCML).” “DCML is a sensory pathway of the central nervous system that conveys sensations of fine touch, vibration, two-point discrimination, and proprioception (position) from the skin and joints.”
- “Retinohypothalamic tract” transmits information on light levels from the “eyes” to the “hypothalamus.” “The retinohypothalamic tract is a photic neural input pathway involved in the circadian rhythms of mammals” [18].

The 7 Majors' neural pathways in the human body that affect cognition

Pathway	Connect	Cognition
Arcuate fasciculus	posterior part of the "tempoparietal junction" with the "frontal cortex" in the brain	Help in cognition connecting areas of the brain involved end the generation and understanding of language, damage of this pathway can cause a form of "aphasia"
Cerebral peduncle	three common areas involved : " brain cortex", "spinal cord" and "cerebellum"	Help in cognition assisting in the refining motor movements, including learning of new motor skills and converting proprioceptive information into balance and posture maintenance.
Corpus callosum	left and right cerebral hemispheres and facilitates interhemispheric communication	Help in cognition transferring motor, sensory, and cognitive information between the brain hemispheres
Pyramidal tracts	consists of two separate tracts in spinal cord: "lateral corticospinal tract" and "medial corticospinal tract"	Leads to the understanding of why the most part, one side of the body is controlled by the opposite side of the brain.
Medial forebrain bundle	is one of the two major pathways connecting the "limbic forebrain", "midbrain", and "hindbrain"	It is a part of the reward system, involved in the integration of reward and pleasure
Dorsal column-medial lemniscus	It is a sensory pathway of the central nervous system	Conveys sensations of fine touch, vibration, two-point discrimination, and proprioception (position) from the skin and joints.
Retinohypothalamic	light levels from eyes to "hypothalamus"	It is involved in the "circadian rhythms" of mammals

FIGURE 2.3 Summary of seven major neural pathways in the human body that affect cognition.

Note*: For more information on other neural pathways, see the "kenhub library" website [19].

A summary of the seven major pathways in the human body that affect "cognition" are shown in Fig. 2.3.

"Studying and analyzing the neural pathways, and their cognitive effects will help to understand, measure, and predict progression to develop effective treatment in neurodegenerative diseases. The development of therapies targeting the next site of the disease will hopefully stop their progression [20]."

2.3.3 Dopamine pathways and cognition

"Dopamine" is one of the most important "neurotransmitters" in the human body that has "many different functions, achieved by traveling to areas in the brain and the human body to transport important information," such as executive thinking, cognition, feelings of reward and pleasure, and voluntary motor movements. These pathways are frequently affected by "neurologic diseases" and they are known as "dopamine pathways." There are eight "dopamine pathways" and the four major ones are "mesocortical," "mesolimbic," "nigrostriatal," and "tuberoinfundibular" [21]:

- "Mesocortical pathway" is a "neural pathway" that connects the "ventral tegmentum" to the "brain cortex," particularly the "frontal lobes," via projections to

the "prefrontal neocortex," "limbic cortex," and "hippocampus." "Mesocortical pathway is essential to the normal cognitive function of the dorsolateral prefrontal cortex (part of the frontal lobe)," and is thought to be involved in motivation, emotional response, poor concentration, and the inability to make decisions; it could be associated with the negative symptoms of schizophrenia, which include avolition (inability to initiate any action), alogia (lack of speech), and flat affect (lack of emotional response).

- "Mesolimbic pathway" connects the following brain structures: "ventral tegmental area," "nucleus accumbens," "amygdala," and "hippocampus":
 - "Ventral Tegmental Area (VTA)" is a part of the "midbrain" and consists of: "dopamine," "GABA," and "glutamate" neurons [22].
 - "Nucleus Accumbens" is in the "ventral striatum" and is composed of "medium spiny neurons" [23]. It is subdivided into "limbic and motor subregions" known as the shell and core. In the "nucleus accumbens" pathway, the "medium spiny neurons" receive input from both the "dopaminergic neurons" of the "VTA" and the "glutamatergic neurons of the hippocampus, amygdala, and medial prefrontal cortex." When they are activated by these inputs, the "medium spiny neurons' projections" release "GABA" (neurotransmitter for producing sleep, reducing anxiety, and forming memory) into the "ventral pallidum."

- The release of “dopamine” in this structure drives the “mesolimbic system.”

 - “Amygdala” is a large nuclear mass in the temporal lobes anterior to the “hippocampus.” “The amygdala has been associated with the assignment of emotions, especially fear and anxiety.”
 - “Hippocampus” is located in the medial portion of the temporal lobes. “It is known for its association with memory.”
- “Nigrostriatal pathway” is a neural pathway that connects the “substantia nigra” with the “striatum.” “Nigrostriatal pathway” is particularly involved in the production of movement, as part of a system called the “basal ganglia motor loop.” “Loss of dopamine neurons in the substantia nigra is one of the main pathological features of ‘Parkinson’s disease,’ leading to a marked reduction in dopamine function in this pathway.” The symptoms of the disease typically do not show themselves until 80%–90% of dopamine function has been lost. These movement disorders may include spasms, contractions, tremors, motor restlessness, parkinsonism, and tardive dyskinesia (irregular/jerky movements) [24].
- “Tuberoinfundibular pathway” is neural pathway that begins in the “arcuate and periventricular nuclei of the hypothalamus,” and project to the “infundibular region of the hypothalamus,” specifically the “median eminence.” “Tuberoinfundibular pathway” malfunction can increase “blood prolactin levels

(hyperprolactinemia)” and produce abnormal lactation, disruptions to the menstrual cycle in women, visual problems, headache, and sexual dysfunction.

A summary of the main dopamine pathways inside the brain that affect the human cognition is shown in Fig. 2.4.

The “neurotransmitter dopamine pathways” play an important role in pleasure/reward, and within the “mesolimbic pathway,” “dopamine” also plays important roles in hormone release, cognition, and movement, as well as affecting the release of “GABA”. Research has shown that “brain neurotransmitter pathway” deficiencies contribute to hundreds of health problems, such as pain, blood sugar problems, immune disorders, digestive dysfunction, movement disorders, anger, depression, anxiety, memory, attention deficits, and many others. “All of these symptoms are reflected in neurologic disorders, and thus again we came to the conclusion that if the neural pathways could be evaluated and measured, we would be able to find new biomarkers for neurologic diseases.”

2.4 Cognitive science

As defined in Section 1.6, “Cognitive science (CoSi) is the interdisciplinary scientific integration that studies the mind and its processes.” It examines the nature, tasks, and the functions of human cognition as the process of acquiring

Four main Dopamine Pathways Inside Brain that affect Cognition

Pathway	Connect	Cognition
Mesocortical	“ventral tegmentum” to the “brain cortex”, particularly the “frontal lobes”, via projections to the “prefrontal neocortex”, “limbic cortex” and “hippocampus”	It is essential to the normal cognitive function of the “dorsolateral prefrontal cortex (part of the frontal lobe)”. It is involved in motivation, emotional response, poor concentration, inability to make decisions, negative symptoms of schizophrenia, which include avolition, alogia and flat affect .
Mesolimbic	“ventral Tegmental Area”, “nucleous Accumbens”, “Amygdala” and “hippocampus”.	Dopamine in this structure drives the mesolimbic system. “Amygdala” has been associated with the assignment of emotions, especially fear and anxiety. “hippocampus” is known for its association with memory. Allows release of neurotransmitter GABA for producing sleep, reducing anxiety, and forming memory
Nigrostriatal	“substantia nigra” with the “striatum”	Involved in the production of movement, as part of a system called the “basal ganglia motor loop”. Loss of dopamine neurons in the “substantia nigra” is one of the main pathological features of “Parkinson’s disease”, leading to a marked reduction in dopamine function in this pathway
Tuberoinfundibular	“arcuate and periventricular nuclei of the hypothalamus”, and project to “median eminence”	It can increase “blood prolactin levels (hyperprolactinemia)” and produce abnormal lactation, disruptions to the menstrual cycle in women, visual problems, headache and sexual dysfunction.

FIGURE 2.4 The four main dopamine pathways inside the brain that affect cognition: mesocortical, mesolimbic, nigrostriatal, and tuberoinfundibular.

knowledge and understanding through thought, experience, and the senses. “CoSi” is the objective of developing theories about humans: perception, action, memory, attention, reasoning, decision-making, language use, and learning. To achieve the integration of these requires the use of many multidisciplinary sciences, such as philosophy, psychology, anthropology, biology, computer science through artificial intelligence, linguistics, neuroscience, education, and mathematics. Their relationship with cognition is shown in Fig. 2.5, and are as follows:

- “Psychology studies the human mind and its functions, while cognitive psychology tries to make functional models of the mind.” Psychologists often define learning as a relatively permanent change in behavior as a result of experience. Then, we can deduce basic models based on: “*trial-and-error learning*,” “*behavioral learning*,” “*insight learning*,” “*cognitive learning*,” and others. Each basic model of learning is described as follows:
 - “*Trail-and-error learning*” is a problem-solving method in which multiple attempts are made to reach a solution. It is a basic method of learning that essentially all organisms use to learn new behaviors. The “*trial-and-error approach*” is used most successfully with simple problems and in games, and it is often the last resort when no apparent rule applies. This does not mean that the approach is inherently careless, for an individual can be methodical in manipulating the variables in an attempt to sort through possibilities which could result in success. Nevertheless, this method is often used by people who have little knowledge in the problem area.

- “*Behavioral learnings*” can be of three major types [25]: *observational*, *classical conditioning*, and *operant conditioning*.
 - “*Observational learning*” is when learning occurs through observations and imitation of others’ behaviors. On this type of learning four actions are essential: *attention*, *motor skills*, *motivation*, and *memory*.
 - “*Classical conditioning*, also known as *Pavlovian* or *respondent conditioning*,” is when learning is based on the association of a previously neutral stimulus and a stimulus that naturally evokes a response.
 - “*Operant conditioning*” is a learning process in which the probability of response occurring is increased or decreased due to reinforcement or punishment.
- “*Insight learning*, also known as *Gestalt theory of learning*,” is when a spontaneous understanding of relationships produces a solution to a problem. This is like a flash of understanding without any process of trial and error.
- “*Cognitive learning*” is learning based on a sequence of process: observing, categorizing, and forming generalizations about the phenomenon. This type of learning leads to “*comprehension*” allowing understanding of the topic and how to fit in other elements, “*memory*” allows the storing of methods and their recall, and its application is in developing problem-solving skills for new situations.
- “*Biology*” studies the living organism and cognition focuses on the survival of organisms and species.

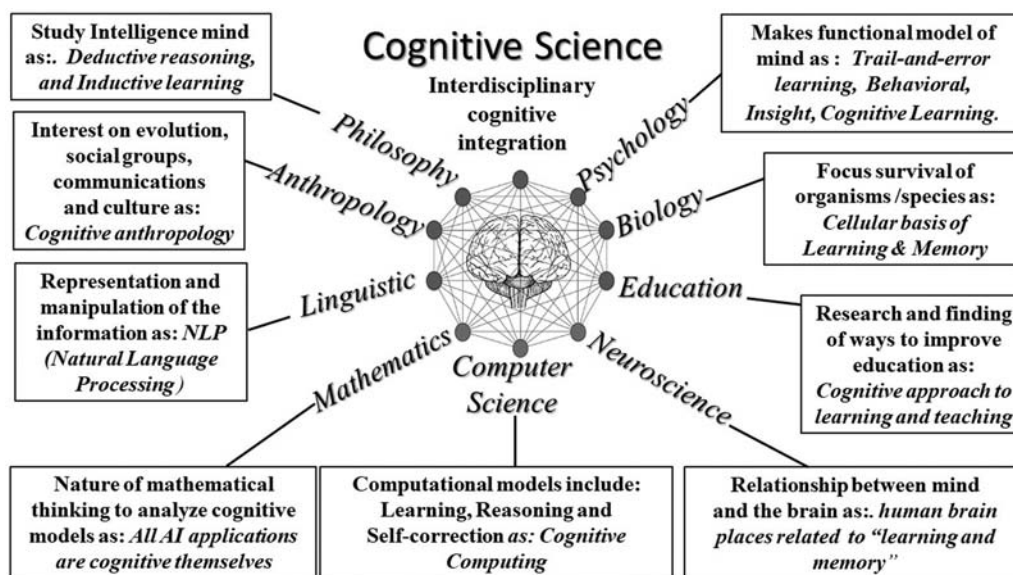


FIGURE 2.5 Cognitive science is an interdisciplinary integration encompassing philosophy, psychology, anthropology, biology, computer science through artificial intelligence, linguistics, neuroscience, education, and mathematics.

There are two theories that explain the cellular basis of “learning and memory”: *nucleotide rearrangement theory* and *cellular modification theory* [26]:

- “Nucleotide rearrangement theory” states that chemical changes in the body are linked to learning. More and harder training results in lower possibilities of forgetting and memory deterioration as their cortical RNA (ribonucleic acid*) increases. However, when RNA synthesis is inhibited, memory becomes impaired.
- Note*: RNA (ribonucleic acid) is a nucleic acid present in all living cells. Its principal role is to act as a messenger carrying instructions from DNA for controlling the synthesis of proteins.
- “Cellular modification theory” focuses on habituation, sensitization, and conditioning in relation to learning and memory. The researchers found that the increase in the release of neurotransmitters results in faster response rates of the sensory-motor neurons synapses. This, in turn, leads to conditioning and sensitization. However, low levels of neurotransmitters result in slower synaptic responses, leading to habituation.
- “Education” is the process of receiving or giving systematic instruction. Whereas in cognitive science it is related to the research and finding of ways to improve education, because “*cognition refers to mental activity including thinking, remembering, learning, and using language.*” When we apply a cognitive approach to learning and teaching, we focus on the understanding of information and concepts. The role of education and intellectual activity on human cognition is very important across our life [27]. Cognitive enrichment early in life may account for some of the variation in cognitive ability in adulthood. Consistently, higher educational attainment is associated with greater levels of cognitive performance. Therefore cognition in adulthood might reflect continued engagement with cognitively complex environments, as well as with a reduced risk of dementia and Alzheimer’s disease [28].
- “Neuroscience” studies the development and function of the nervous system, which includes the brain, spinal cord, and nerve cells throughout the body, whereas “*cognition covers the relationship between the mind and the brain.*” In the human brain there are two places directly related to “learning and memory”: the “hippocampus and mediodorsal thalamus.”
 - “Hippocampus” is situated in the medial temporal lobe (one in each hemisphere) and is responsible for the consolidation of “short-term memory” and “long-term memory,” in particular, the formation of new memories related to experiences and events known as “episodic memories.” “declarative memories” are those that can be verbalized more explicitly than episodic memories, and are formed but not stored in the

“hippocampus,” see Fig. 2.2B; these memories as well as past events are believed to be stored in the “frontal and temporal lobes.” In addition, a process called “long-term potentiation (LTP)” refers to the increase in neural responsivity occurring in the hippocampus. “LPT” is involved in “spatial learning” that encodes information about the environment to facilitate navigation through space and recalls the location of motivationally relevant stimuli.

- “Mediodorsal thalamus” in the center of the brain is a large nucleus in the thalamus, as shown Fig. 2.2B, and has a role in memory and other “cognitive tasks.”
- “Computer Science through Artificial Intelligence” focuses on obtaining “computational AI models using process that include learning, reasoning and self-correction.” AI applies algorithms that have the ability to automatically learn and improve from experience without being explicitly programmed; in other words, the “AI applications are cognitive by themselves.”
- “Mathematics” is the abstract science of number, quantity, and space, whereas “cognitive science can explain the nature of mathematical thinking to analyze AI cognitive models.”

“The magic of AI is based specially on: **mathematics** that is used to define each AI model, **statistics** that help in the criteria for validate if an AI model is accepted or not, and **data mining** to extract important information of the databases, and all AI applications are cognitive by themselves.”

- “Linguistics” uses language to represent information, and “*cognition is related through the representation and manipulation of the information.*” Cognition has been evolved in Natural Language Processing (NLP) in two different ways: *Natural Language Understanding* and *Natural Language Generation*:
 - *Natural Language Understanding (NLU)** is what a computer would need to do, if it were to interpret a spoken human command and act on it.
 - Note*: On AI, “NLU” is known as “speech recognition,” as indicated in Fig. 1.13 in the “Proposed General Architecture of a Cognitive Computing Agents System (AI-CCAS).”
 - *Natural Language Generation (NLG)** involves taking a formal symbolic representation of an idea and converting it to an expression in English or some other natural language.
 - Note*: On AI, NLG is known as “speech generation,” as indicated in Fig. 1.13 in the “Proposed General Architecture of a Cognitive Computing Agents System (AI-CCAS).”
- *Anthropology* studies the human societies, their cultures, and their development. It is complemented by

the interest of cognition on evolution, social groups, communication, and culture. “*Cognitive anthropology focuses on what people in different groups know, how this implicit knowledge changes people’s perception of the world around them, and establishes an association with the world.*” It is much used today in today’s developed internet to introduce many new technologies based on “*human cognition*” to cultivate interest in new devices of computer technology [29].

- *Philosophy* studies the fundamental nature of knowledge, reality, and existence, while “*cognition study the intelligence mind.*” Thus we can establish that the primary method of philosophical inquiry is “*reasoning,*” and it could be “*deductive*” or “*inductive*”:
 - “*Deductive reasoning*” is based on the application of rules of logic to a statement about we want to learn; for example, to learn a language, we start learning the rules, then examples, and finally practice them.
 - “*Inductive learning*” starts with examples, as learners concentrate on finding rules based on the detection of patterns, then we practice testing the rules deduced.

A summary of the multidisciplines that are integrated within cognitive science and their examples with regard to human cognition is shown in Fig. 2.5.

We focus in the study of interactions of linguistics, neurology, mathematics, and computer science with cognitive science in biomedical engineering problems, to find models to analyze, detect, classify, and forecast the process of different illness and injuries of the human body with special emphasis on neurologic disorders applying: Artificial Intelligence (AI) through Machine Learning (ML), Deep Learning (DL), and Cognitive Computing (CC).

2.5 Natural Language Processing

“*Linguistics*” is the scientific study of language and involves analyzing language form, its meaning and context. It has evolved in cognition as “*Natural Language Processing (NLP).*” “*NLP*” is a branch of “*AI technology*” used to aid computers to understand human’s natural language. Its main objectives are to read or hear, understand, analyze, manipulate, and generate as text or speech human language or vice versa through the application of “*AI algorithms.*”

“*NLP*” has many applications, such as information retrieval, information extraction, language translation, language spelling, and grammatical accuracy of texts in text processors, text simplification, interactive voice response (IVR), sentiment analysis, text summarization, computer personal assistant, spam filters, speech recognition, natural language generation, speech generation, and many others.

- “*Information retrieval*” is used by internet search engines, such as Google, to find relevant and similar results.
- “*Information extraction*” is used in eBooks or emails for analyzing their structure to extract useful text, for example, Gmail, Outlook, and others.
- “*Language translation*” is used to translate from one language to other, for example, “*Google Translate*” and others.
- “*Language spelling and grammatically accuracy of texts in text processors,*” for example, “*Microsoft Word,*” and others.
- “*Text simplification*” is used to simplify the meaning of a sentence, for example, “*Rewordify,*” and others.
- “*Interactive voice response (IVR)*” is used by call centers to respond to certain users’ requests.
- “*Sentiment analysis*” is used to analyze the general sentiment of the user, for example, “*Hater News,*” and others.
- “*Text summarization*” is used to give a summary of sentences, for example, “*Smmry*” or “*Reddit’s autoldr,*” and others.
- “*Computer personal assistant*” is used to facilitate the search on the web or computer systems, for example, “*Siri,*” “*Cortana,*” “*Alexa,*” and others.
- “*Spam filters*” are used to avoid the reading of unwanted emails, for example, “*Gmail,*” “*Outlook*” spam filters.
- “*Speech recognition*” is used to convert speech to text, for example, “*Google WebSpeech,*” “*Vocal ware,*” and others.
- “*Natural language generation*” is used to generate text from image or video data, for example, “*Arria NLG studio,*” and others.
- “*Speech generation*” is used to convert text-to-speech, for example, “*Cloud Text-to-speech,*” and others.

There are many AI methods that can be used in “*NLP,*” a typical algorithm to classify text is shown in Fig. 2.6. These steps are: *reading dataset, preprocessing text, data vectorization, feature engineering, and ML model selection.*

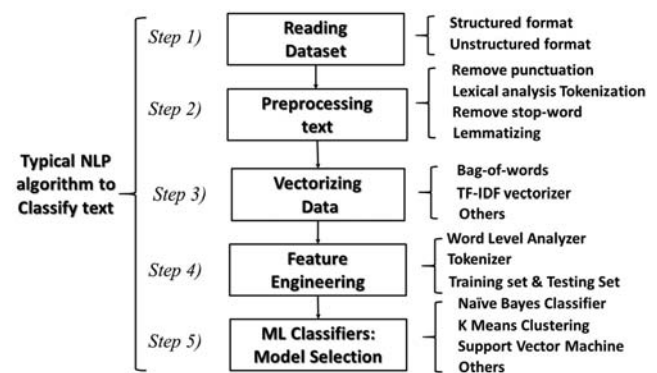


FIGURE 2.6 Typical NLP algorithm to classify text.

2.5.1 Step 1) Reading dataset for NLP

Reading Dataset of text can be done in two forms: “structured and unstructured data format.” A “structured data” has very well-defined patterns and usually comes from a database format; meanwhile “unstructured data” does not have a proper structure.

2.5.2 Step 2) Preprocessing text for NLP

Preprocessing text is made in four substeps: “remove punctuation,” “tokenization,” “remove stop-words,” and “lemmatizing,” where:

- “Remove punctuation,” the punctuation provides grammatical context to a sentence for the human understanding; but “NLP uses a vectorized method which counts the number of words, so the removal of special characters is necessary,” for example, “What’s your name?” is converted to “whats your name.”
- “Tokenization” is a “Lexical Analyzer method” that separates text into units as sentences or words, this give a structure to unstructured text, for example, “What’s your name?” is converted first to “what your name” when the punctuation is removed, then to “what, your, name” during the “tokenization.”
- “Remove stop-words” is when common words like “been,” “or,” “and” etc., which appear in any text but do not tell us much about the data, are removed, for example, “I’ve been searching for the answer” is converted to “ive been searching for the answer” when the punctuation is removed, then to “ive, been, searching, for, the, answer” during the “tokenization,” and

finally is converted to “ive, searching, answer” after the “removal of stop-words.”

- “Lemmatizing” is a method to find the canonical form of a word, it uses a dictionary-based approach, that is, “I’ve been searching for the answer” is converted to “ive been searching for the answer” when the punctuation is removed, then to “ive, been, searching, for, the, answer” after the “tokenization,” and then it is converted to “ive, searching, answer” after the “remove of stop-words,” and finally to “ive, search, answer” during the “lemmatizing step.”

2.5.3 Step 3) Data vectorization in NLP

“Data vectorization” is a method to encode text as integers, with the purpose of obtaining a numeric form that allows the creation of future vectors that can be easily understand for the “ML to understand the text data.” Some frequently used methods for data vectorization in “NLP” are known as: “Bag-of-Words or Count-vectorizer” and “TF-IDF vectorizer.”

- “Bag-of-Words or Count-vectorizer” is based on: count and frequencies of words, where:
 - Count of words also known as Total Number of Terms (TN) is the times each word appears in a document.
 - Frequency of words (FW) is the number that each word appears in the document.

The “Bag-of-Words NLP method” is explained with an example in [Table 2.1](#).

TABLE 2.1 Examples of the Bag-of-Words NLP method.

We have four sentences to analyze using the Bag-of-Words NLP method:

“He was the best of us”

“He was the worst of us”

“He was the older of us”

“He was the fastest of us”

They are going to be analyzed as four different documents, the “count of words” on the total of documents are 9; these are:

‘He’, ‘was’, ‘the’, ‘best’, ‘of’, ‘us’, ‘worst’, ‘older’, ‘fastest’

Then, we create the vectors based on the “frequency of words” from the 9 unique words in each document, such as in the first: “He was the best of us”:

“He” = 1, “was” = 1, “the” = 1, “best” = 1, “of” = 1, “us” = 1, “worst” = 0, “older” = 0, “fastest” = 0

And, finally the generated vectors for each document are:

“He was the best of us” = [1, 1, 1, 1, 1, 1, 0, 0, 0]

“He was the worst of us” = [1, 1, 1, 0, 1, 1, 1, 0, 0]

“He was the older of us” = [1, 1, 1, 0, 1, 1, 0, 1, 0]

“It was the age of foolishness” = [1, 1, 1, 0, 1, 1, 0, 0, 1]

In this simple example each word is known as “token” and is called a “gram” in “NLP.” Sometimes it is necessary to work with vocabulary of two-word pairs known as the “Bigram Model.”

For example, the bigrams for the first document: “It was the best of time” are:

“it was,” “was the,” “the best,” “best of,” “of times”

Example 2.1: Theoretical example of Bag-of-Words NLP method

- “*TF-IDF vectorizer*” is another “*NLP*” method based on the calculation of: “*Term Frequency* and *Inverse Document Frequency (IDF)*.” Their meanings are described in the following equations:
 - “*Term Frequency (TF)*” is a number that represents the “*scoring of the frequency of words in each document*,” and it can be calculated as shown in Eq. (2.1).

$$\text{Term Frequency } TF(i,j) = \frac{FW(i)}{TL(j)} \quad (2.1)$$

where FW = Frequency of words, and TL = Total Number of words.

- *Inverse Document Frequency (IDF)* is a statistics number that “*indicates how important the word is across documents*,” and it is indicated in Eq. (2.2).

$$\text{Inverse Document frequency } IDF(i) = \ln\left(\frac{TND}{Tt(i)}\right)$$

Note to the editor: Please align this equation

with the indentation of the last sentence (2.2)

where TD = total number of documents, and Tt = Number of document with the word “*i*” in it.

The TF-IDF vectorizer is calculated as shown in Eq. (2.3).

$$\mathbf{TF-IDF} \text{ vectorizer } \mathbf{TF-IDF} = \mathbf{TF} \times \mathbf{IDF}$$

Note to the editor: Please align this equation

with the indentation of the last sentence (2.3)

The result of the **TF-IDF** vectorizer is a matrix, as represented in Table 2.2.

2.5.4 Step 4) Feature engineering in NLP

“*Feature engineering*” consists of deciding feature creation as the selection for the best granularity for the “*vectorizer*.” Frequent alternatives are: “*Word Level Analyzer*” and “*Tokenizer*.”

- “*Word Level Analyzer*” assigns each word to its own terms. It works very well with colloquial English documents, such as URL and emails.

- “*Tokenizer*” assigns to each word its own term: this method is able to extract more information than word analyzers; it splits documents into “*tokens*,” based on white space and special characters, that is, “*what’s next*” might be split into: “*what’s, next*.”

Before training the vectorizer, one should split the data into a “*training set*” and “*testing set*.”

Note*: *Frequently the “testing set” is 10% of the total data.*

2.5.5 Step 5) ML model selection for NLP

“*ML model selection*” consists of selecting the type of classifier to use between several candidates for “*ML classifiers*” and evaluating them against the testing set to deduce which one works best. The eight ML algorithms most frequently used are*: *Naive Bayes Classifier*, *K Means Clustering*, *Support Vector Machine*, *Artificial Neural Networks*, *Latent Dirichlet Allocation*, *Random Forests*, *Decision Trees*, and *Nearest Neighbors*.

Note*: *These ML classifiers will be studied in Chapter 4, Machine Learning Models Applied to Biomedical Engineering, and Chapter 5, Deep Learning Models Principles Applied to Biomedical Engineering.*

2.6 MATLAB[®] toolboxes solution for natural language processing

“*MATLAB*” is an engineering numerical computing environment and proprietary programming language developed by MathWorks. “*MATLAB*” allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including “*C*,” “*C++*,” “*C#*,” “*Java*,” “*Fortran*,” and “*Python*.” The basic “*MATLAB*” is augmented with “*toolboxes*,” these optional additions give great advances to MATLAB offering specialized powerful functions in many different fields. Some of the “*MATLAB toolboxes*” that will be used in this book for examples and exercises are: “*Statistics and Machine Learning*,” “*Deep Learning*,” “*Text Analytics*,” “*Fuzzy Logic*,” “*Computer Vision*,”

TABLE 2.2 The matrix result of TF-IDF vectorizer NLP method.

	Document 1	Document 2	...	Document j
Term 1	[TF-IDF](1,1)	[TF-IDF](1,2)	...	TF-IDF(1,j)
Term 2	TF-IDF(2,1)	TF-IDF(2,2)	...	TF-IDF(2,j)
...
Term i	TF-IDF(i,1)	TF-IDF(i,2)	...	TF-IDF(i,j)

“Image Processing,” and “Audio.” Where their general descriptions are:

- “Statistics and Machine Learning Toolbox” provides functions and apps to describe, analyze, and model data. It allows descriptive statistics and plots for exploratory data analysis, to fit probability distributions to data, to generate random numbers for simulations, and to perform hypothesis tests. Regression and classification algorithms allow the drawing of inferences from data and the building of predictive models. The toolbox provides “supervised and unsupervised machine learning algorithms.”
- “Deep Learning Toolbox (formerly Neural Network Toolbox)” provides a framework for designing and implementing “deep neural networks with algorithms,” “pretrained DL models,” and apps. It allows “convolutional neural networks (ConvNets, CNNs)” and “long short-term memory (LSTM) networks” to perform classification and regression on images, time-series, and text data. Apps and plots help you to visualize activations, edit network architectures, and monitor training progress.
- “Text Analytics Toolbox” provides algorithms and visualizations for preprocessing, analyzing, and “modeling text data for NLP applications.” Models created with the toolbox can be used in applications such as sentiment analysis, predictive maintenance, and topic modeling.
- “Fuzzy Logic Toolbox” provides functions, apps, and a Simulink block for analyzing, designing, and simulating systems based on “fuzzy logic.” The product guides you through the steps of designing “fuzzy inference systems (FIS).” Functions are provided for many common methods, including “fuzzy clustering” and “adaptive neurofuzzy learning.” The toolbox lets you model complex system behaviors using simple logic rules, and then implement these rules in a “FIS.” Besides, you can use it as a “stand-alone fuzzy inference engine.”
- “Computer Vision Toolbox” provides algorithms, functions, and apps for designing and testing computer vision, 3D vision, and video processing systems. It performs “object detection and tracking,” as well as “feature detection,” “extraction,” and “matching.”
- “Image Processing Toolbox” provides a comprehensive set of reference-standard algorithms and workflow apps for “image processing,” “analysis,” “visualization,” and “algorithm development.” It can perform “image segmentation,” “image enhancement,” “noise reduction,” “geometric transformations,” “image registration,” and “3D image processing.”
- “Audio Toolbox” provides tools for “audio processing,” “speech analysis,” and “acoustic measurement.” It includes algorithms for “audio signal processing” such as equalization and dynamic range control and

“acoustic measurement” such as impulse response estimation, octave filtering, and perceptual weighting. It also provides “algorithms for audio and speech feature extraction” and “audio signal transformation.”

- “DSP System Toolbox” allows the “design and simulation of streaming signal processing systems.” It provides algorithms, apps, and scopes for designing, simulating, and analyzing signal processing systems in MATLAB and Simulink. You can “model real-time DSP systems for communications,” “radar,” “audio,” “medical devices,” “IoT,” and other applications.
- Many others “MATLAB Toolboxes” are useful for “AI.”

2.6.1 Natural Language Processing applications with MATLAB

The MATLAB AI solutions to apply “NLP” allow the creation of many applications, such as:

- “Automatic recognition of spoken commands for computer/robots’ smart medical systems.”
- “Identify people speech using pitch” and “Mel-frequency cepstral coefficients (MFCC).”
- “Denoise speech using DL networks.”
- Classify gender of a speaker using DL algorithms, such as “LSTM” and others.
- “Speech emotion recognition.”
- Customer care calls.
- Virtual assistants.
- Machine translation and dictation.

There are many more “NLP” useful application for “human voice cognitive analysis for cognitive services, such as voice therapy” and others.

As explained in Section 2.5, “NLP is a branch of AI.” It is the technology used to “aid computers to understand human’s natural language. Its main objectives are to read or hear, understand, analyze, manipulate, and generate as text or speech human language” through the application of “AI algorithms.”

“NLP” can be defined in a practical way as the broad class of computational techniques for incorporating speech and text data, along with other types of engineering data, into the development of “AI systems.” “NLP” can be used to combine and simplify raw human language sources that come from audio signals, audio documents, audio databases, audio web, and audio social media; with the objectives of insight visualization, application of text mining, obtaining models, and obtaining classification using “ML algorithms” for different purposes, such as:

- Deduct “Topic Modeling*”
Note*: See Section 2.6.2.1, MATLAB: Case for research: “NLP Topic Models.”
- Obtain “Sentiments analysis” of topics that could be positive or negative,

- “Labels and tagging automation” of speech recording,
- Detect and apply “Voice command.”

And many applications for health, finance, manufacturing, information technology, and other industries.

Actually, “NLP is applied in MATLAB using three different toolboxes”: “Text Analytics Toolbox,” “Audio Toolbox,” and “Statistics and Machine Learning Toolbox”.

2.6.2 “NLP Topic Models” with MATLAB

Analyzing streaming text or static text data with “NLP Topic Models” are extremely useful to detect “trends,” “sentiments,” and others cognitive behaviors. They can be used in many “Biomedical Engineering applications,” such as safety medical records, medical research, sentiment analysis, cybersecurity, etc. Actually, the most common “AI algorithms used for NLP” are: “Recurrent Neural Networks (RNNs),” “Linear Regression,” “Support Vector Machine (SVMs),” “Naive Bayes Classifier,” “Latent Dirichlet Allocation (LDA),” “Latent Semantic Analysis,” and “word2vec.” The typical approach is the use of “ML algorithms,” which are going to be studied in Chapter 4, Machine Learning Models Applied to Biomedical Engineering,” and Chapter 5, Deep Learning Models Principles Applied to Biomedical Engineering, of this book.

As explained in Section 2.5, an “NLP application for classifying text” is “NLP Topic Models” and the typical algorithm is shown in Fig. 2.6. The necessary steps are: Reading dataset, Preprocessing text, Data vectorization, Feature engineering, and ML model selection. All of them are applied in sequence in the MATLAB program shown in Table 2.3.

2.6.2.1 Research 2.1: “NLP Topic Models between a blog with a patient with neurologic disease and a researcher”

2.6.2.1.1 General objective

“Analyze a text chat conversation between a patient “x” with neurologic disease and a biomedical researcher to obtain NLP Topic Models from the conversation to detect the four main topics.”

2.6.2.1.2 Specific objectives of this research

1. Find the most four important topics of all the dialog conversation
2. Apply “Bag-of-Words method” for “vectorization,” “LDA model” for classification and obtain “word-cloud chart” for the four most important topics of the dialog conversation.

3. Create a chart for “Mixtures of the four main topics in each document created.”
4. Analyze text chat conversation using: “Phrases of three consecutive words.”

2.6.2.1.3 Developing a MATLAB program for this research: “NLP Topic Models”

The MATLAB program* following the basic algorithm shown in Fig. 2.6 is applying the specific objectives for this research, and the MATLAB program is shown in Table 2.3.

Note*: For this example, it is necessary to install the following MATLAB toolboxes: “Statistics and Machine Learning Toolbox” and “Text Analytics Toolbox.”

The program uses MATLAB function named “preprocessText(),” for preprocessing text applying the following techniques: “tokenization,” “lemmatizing,” “erase punctuation,” and “remove words” ≤ 2 characters or remove words ≥ 15 characters, as explained in Section 2.5; the function is indicated in Table 2.4.

2.6.2.1.4 Results from the MATLAB program for research: “NLP Topic Models”

When the MATLAB program from Table 2.3 is run, we obtain the following results:

1. The results for Step 1 show that the dataset is structured in three fields: “Time,” “From,” and “Event_Narrative,” as indicated in Fig. 2.7A.
2. The content of the variable dataText(i) is an array of 10×1 string that contains the text chat conversation as indicated in Fig. 2.7B.
3. The results for Step 3) show that the “Bag-of-Words” has been processed and the document tokenized as indicated in Fig. 2.7C.
4. In Step 4) the value for the “numTopics” variable is for the number of topics that minimize the amount of the numbers of them, and another criterion is that models fit with larger numbers of topics may take longer to converge. The results for Step 5) of Fit an “LDA model” based on the four topics in the four documents using “wordcloud” chart are shown in Fig. 2.8. The main topics are “symptoms (multiple diagnosis),” “disease (sclerosis),” “nervous system (brain, body analyze),” and “movement (human disorder).”
5. The results for the special Step 6) indicate a chart of the probability of finding the following vocabulary “Neurologic diseases measuring signals from body movements” in each topic. The result is stored in the vector variable “topicMixture = [0.2384,0.2059, 0.4084,0.1472]” as shown in Fig. 2.9A.
6. The results for special Step 7) are the visualization of the multiple topic mixtures in the four documents as shown in Fig. 2.9B.

TABLE 2.3 MATLAB Program for “NLP Topic Models,” between a patient with neurologic disease and a researcher to analyze a text chat conversation.

```

%% Section 2.5.1.2 NLP Example of Topic Models as shown in Fig. 2.6
% Textbook: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
%% Environment variables
rng('default'); % default values to MATLAB random generator
%% Step 1) Reading dataset
data = readtable("messages.csv", 'TextType', 'string'); % Structured datafile
% In subdirectory. \Exercises_book_ABME\CH2\NLP example of Topic models
head(data)
textData = data.event_narrative; % concentrate on
textData(1:9) % 9 text dialog conversation
%% Step 2) Preprocessing text: Tokenization, Lemmatizing, Erase punctuation,
% Remove list of stop words, Remove words with >= 2 Charac, with >= 15
documents = preprocessText(textData);
documents(1:4)
%% Step 3) Vectorizing Data
bag = bagOfWords(documents)
documents = preprocessText(textData);
documents(1:4)
%% Step 4) Feature Engineering
numTopics = 4;
%% Step 5) ML Classifier
% A Latent Dirichlet Allocation (LDA) is a topic model which discovers
% underlying topics in a collection of documents and infers the word
% probabilities in topics.
mdl = fitLda(bag, numTopics, 'Verbose', 0);
figure; % Visualize Topics Using Word Clouds function
for topicIdx = 1:4
    i) subplot(2,2,topicIdx)
    ii) wordcloud(mdl,topicIdx);
    iii) title("Topic" + topicIdx)
end
%% Special step 6) View Mixtures of Topics in Documents
newDocument = tokenizedDocument("Neurologic diseases measuring signals from body movements");
% Use transform to transform the documents into vectors of topic probabilities.
topicMixture = transform(mdl,newDocument);
figure
bar(topicMixture)
xlabel("Topic Index")
ylabel("Probability")
title("Document Topic Probabilities")
%% Special step 7) Visualize multiple topic mixtures of documents
figure
topicMixtures = transform(mdl,documents(1:4));
barh(topicMixtures(1:4,:), 'stacked')
xlim([0 1])
title("Topic Mixtures")
xlabel("Topic Probability")
ylabel("Document")
legend("Topic" + string(1:numTopics), 'Location', 'northeastoutside')
%% Special step 8) Analyze text chat conversation using phrases of 3 consecutive words.
bag = bagOfNgrams(documents, 'NGramLengths', 3);
figure
wordcloud(bag);
title("Neurologic diseases")

```

Note: This MATLAB program can be downloaded from the book's website companion and installed in the following path "...!Exercises_book_ABME\CH2\NLP example of Topic models\TopicModels.m".

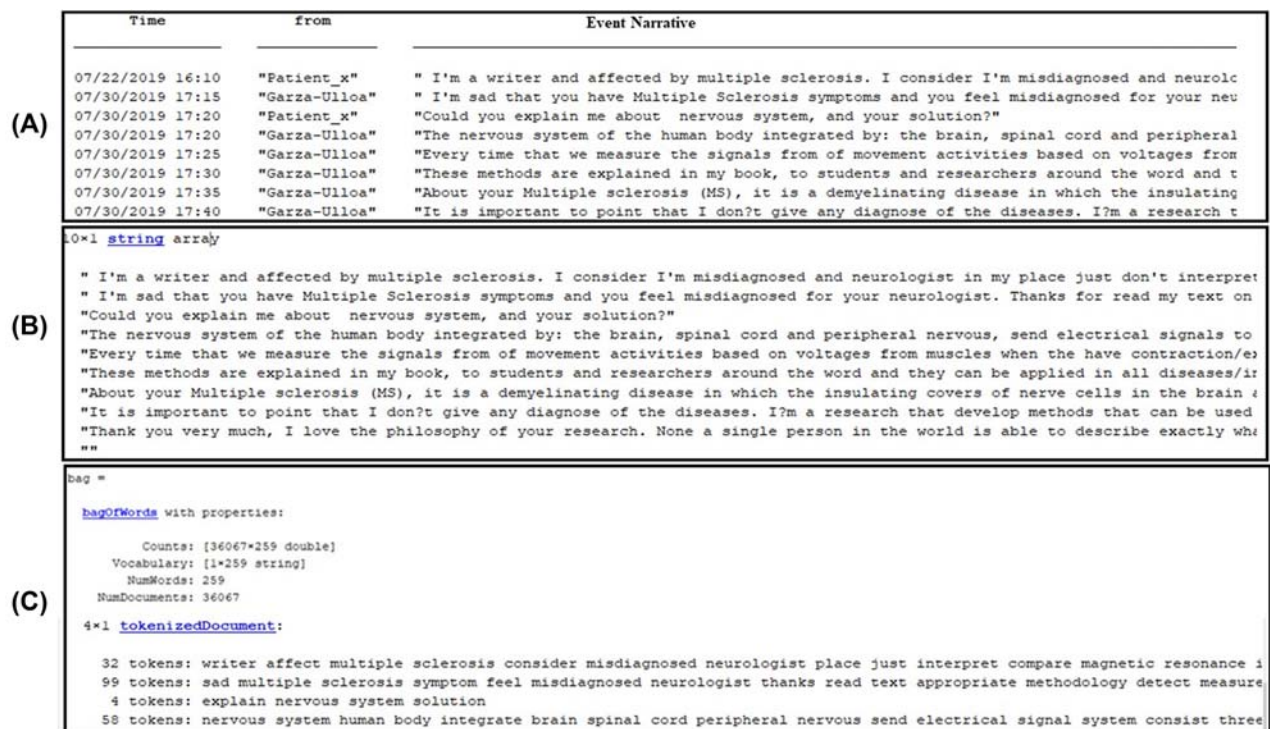
TABLE 2.4 The MATLAB main program shown at Table 2.3 for “NLP Topic Models” call the function shown at table 2.4 for preprocessing text as needed on NLP.

```

function documents = preprocessText(textData)
documents = tokenizedDocument(textData);           % Tokenization
documents = addPartOfSpeechDetails(documents);     % Lemmatizing
documents = normalizeWords(documents, 'Style', 'lemma');
documents = erasePunctuation(documents);          % Erase punctuation
documents = removeStopWords(documents);           % Remove stop-words
documents = removeShortWords(documents, 2);       % Remove words <= 2 charac
documents = removeLongWords(documents, 15);      % Remove words >= 15 charac
end

```

Note: This function can be downloaded from the website companion and install in the following path “...\\Exercises_book_ABME\\CH2\\NLP example of Topic models\\preprocessText.m”.

**FIGURE 2.7** MATLAB text results for “NLP Topic Models”: (A) dataset from conversation, (B) variable dataText(i), and (C) “Bag-of-Words” results.

7. The special Step 8) for analyzing text chat conversation using phrases of three consecutive words can be obtained using the MATLAB function “bagofNgrams,” that allow specify the number of consecutives words to analyze based on: “gram” for one word, “bigram” for two consecutive words, etc. The “bagofNgrams” objects can be used in other “Text Analytics toolbox functions,” such as “wordcloud” and “fitlda.” The results for this step are shown in Fig. 2.10A the objects created by “bagofNgrams” and in Fig. 2.10B the

visualization in a chart of “bagofNgrams” for a trigram or three consecutive words.

2.6.2.1.5 Conclusions and recommendation

The example of “NLP Topic Models with MATLAB” shows that by using “AI NLP” analysis that we have powerful commands available on the “Text Analytics Toolbox,” that allow the detection of the main topics in different “grams.” We can divide the function available

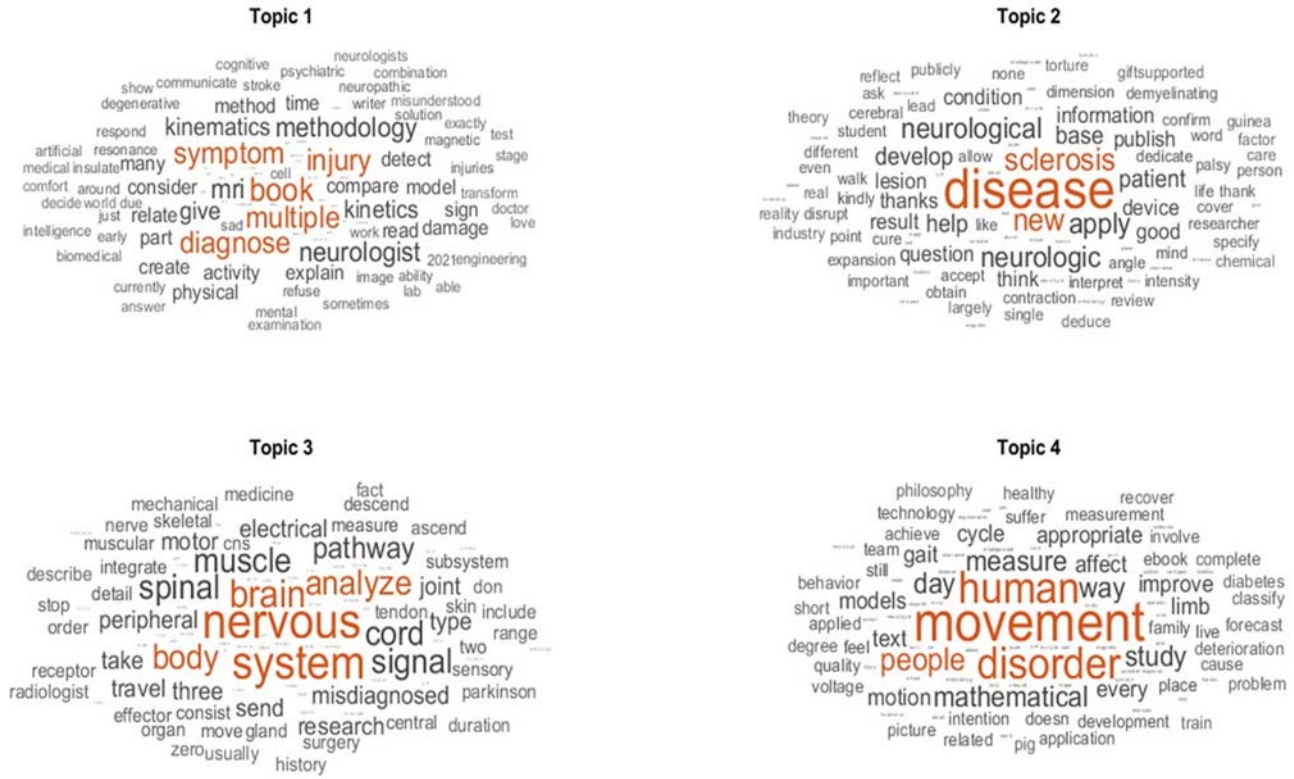


FIGURE 2.8 MATLAB chart results for “NLP Topic Models” showing the four main topics: symptoms (multiple diagnosis), disease (sclerosis), nervous system (brain, body analyze), and movement (human disorder).

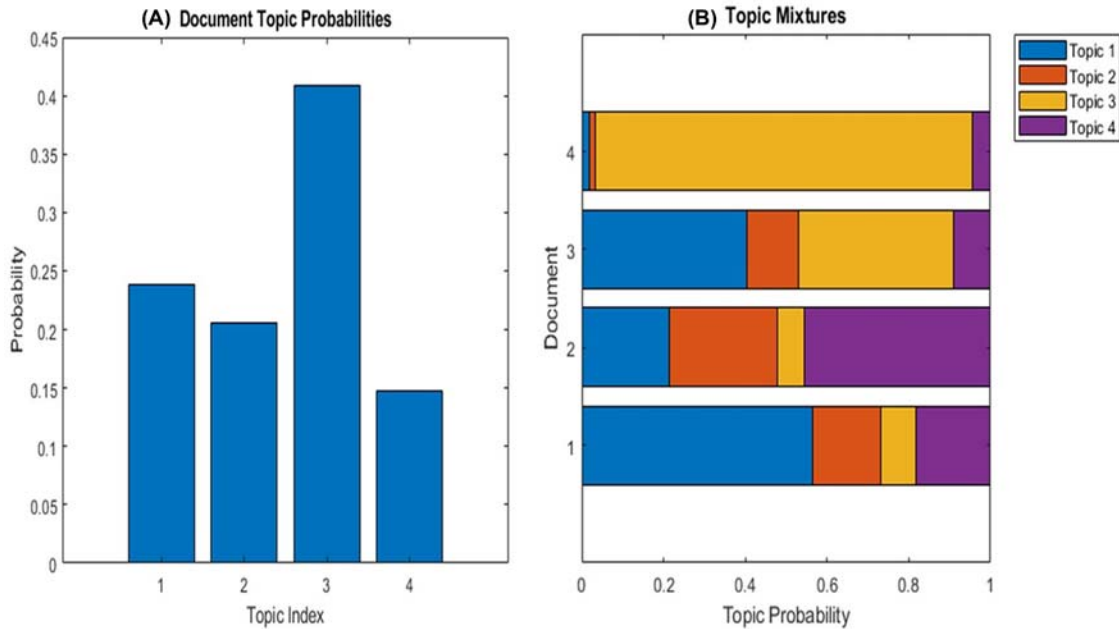


FIGURE 2.9 MATLAB chart results for “NLP Topic Models”: (A) Document Topic probabilities for “Neurologic diseases measuring signals from body movements” and (B) Topic Mixtures.

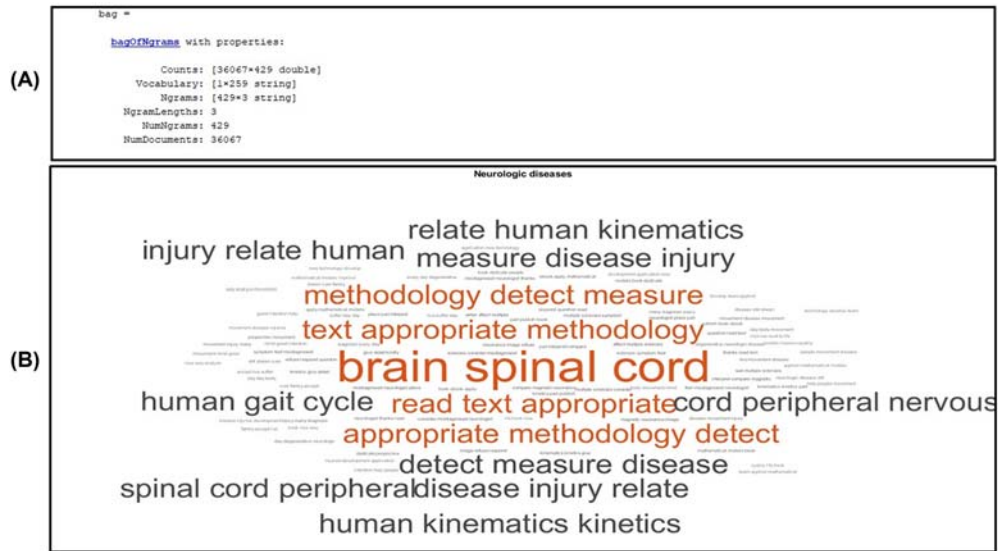


FIGURE 2.10 MATLAB “NLP Topic Models of three consecutive words”: (A) text result and (B) chart result: brain spinal cord.

on this toolbox into *Visualization function, model & predict function, import function, preprocess function, and string functions*, where:

- *Visualization functions* are used for word clouds and text scatter plots to summarize and validate results, as follows:
 - “wordcloud” is used to create word cloud chart from Bag-of-Words or LDA model.
 - “wordCloudCounts” counts words for word cloud creation.
 - “textscatter” is used for a 2D scatter plot of text.
 - “textscatter3” is used for a 3D scatter plot of text.
 - “heatmap” can create a heatmap chart.
 - “histcounts” is used for histogram bin counts.
 - “discretize” is used for group data into bins or categories.
- *Model & Predict functions* allow conversion of text into numeric representation using Bag-of-Words or pretrained word embedding models, and apply specialized ML algorithms for prediction and topic modeling as follows:
 - “readWordEmbedding” for reading word embedding from text file. “trainWordEmbedding” to train word embedding.
 - “word2vec/vec2word” to map words to embedding vectors.
 - “ldaModel” for a Latent Dirichlet allocation (LDA) model.
 - “lsaModel” for a Latent semantic analysis (LSA) model.
 - “bagOfWords” for Bag-of-Words model.
 - “fitlda” to fit latent Dirichlet allocation (LDA) model.
 - “fitlsa” to fit a latent semantic analysis (LSA) model.
 - “predict” to predict top LDA topics of documents.
 - “fitdist” to fit probability distribution object to data.
 - “fitrlinear” to fit linear regression model to high-dimensional data.
 - “fitllinear” to fit linear classification model to high-dimensional data.
 - “fitcecoc” to fit multiclass models for classifiers.
- *Import functions* used to extract text from Microsoft Word files, PDFs, text files, and spreadsheets as follows:
 - “extractFileText” to read from PDF, Microsoft Word, and plain text.
 - “textscan” to read formatted data from text file or string.
 - “readtable” to create table from file.
 - “compose” to convert data into formatted string array.
 - “xlsread” to read from Microsoft Excel spreadsheet file.
 - “webread” to read content from RESTful web service.
 - “TabularTextDatastore” a datastore for tabular text files.
 - “FileDatastore” a datastore with custom file reader.
 - “SpreadsheetDatastore” a datastore for spreadsheet files.
- *Preprocess functions* to remove less helpful artifacts such as common words, punctuation, and URLs and apply text normalization to stem words to their root word as follows:
 - “tokenizedDocument” to split documents into collections of words.

- “*normalizeWords*” to remove inflections from words using the Porter stemmer.
- “*bagOfWords*” for a Bag-of-Words model.
- “*stopWords*” for stop word list.
- “*context*” to search documents for word occurrences in context.
- “*removeWords*” to remove selected words from document or Bag-of-Words.
- “*removeLongWords*” to remove long words from documents or Bag-of-Words.
- “*removeShortWords*” to remove short words from documents or Bag-of-Words.
- “*removeInfrequentWords*” to remove words with low counts from Bag-of-Words model.
- “*erasePunctuation*” to erase punctuation from text and documents.
- *String functions* for manipulate, compare, and store text data efficiently as follows:
 - *str* = declare a string variable, that is, “*Hello, world.*”
 - *str* = declare a string array, that is, [*“Hello”, “World”*].
 - “*str = string(C)*” to convert a character vector *C* to a string.
 - “*str2double*” to convert a string to double numbers.
 - “*strlen*” to return the length of strings.
 - “*isstring*” to determine if input is string array.
 - “*join*” to combine strings.
 - “*split*” to split strings in string array.
 - “*splitlines*” to split string at newline characters.
 - “*replace*” to find and replace substrings in string array.
 - “*contains*” to determine if pattern is in string.
 - “*erase*” to delete substrings within strings.
 - “*extractBetween*” to extract substrings between indicators.
 - “*extractAfter*” to extract substring after specified position.
 - “*extractBefore*” to extract substring before specified position.
 - “*strcmp*” to compare strings.
 - “*regexp*” to match regular expression (case sensitive).

2.6.3 “NLP audio files” with MATLAB

MATLAB provides a few built-in functions that allow one to import and export audio files. Newer versions of MATLAB that include “*Audio Toolbox*,” the functions “*audioread*” and “*audiowrite*” can be used to read and write data to/from various types of audio files.

2.6.3.1 Research 2.2: “NLP read and reproduce audio files stored and by frame in real time using MATLAB”

2.6.3.1.1 General objective

“*Reproduce audio files as entire file stored and frame-by-frame as in real time in MATLAB.*”

2.6.3.1.2 Specific objectives of this research

1. “*Read entire audio file*” into workspace and then send to the computer speakers.
2. “*Read each frame of audio*” into workspace and then send audio by frame to speakers.
3. “*Release resources*” used to reproduce audio files.

2.6.3.1.3 Developing a MATLAB program for this research: “NLP MATLAB audio files”

The MATLAB program is shown in [Table 2.5](#). For this example, the MATLAB toolboxes “*Audio Toolbox*” and “*DSP System Toolbox*” must be installed.

2.6.3.1.4 Results of the MATLAB program for “NLP read audio files”

When the program shown in [Table 2.5](#) is run, two text messages are shown on the screen as indicated in [Table 2.6](#).

After the first message “*Reading entire audio file...*” is shown on the screen, the audio message is heard on the speaker of your computer. This is achieved by two MATLAB commands from the “*Audio Toolbox*”:

- “[*Y, FS*] = *audioread(FILENAME)*,” it reads an audio file specified by the character vector or string scalar *FILENAME*, returning the sampled data in *Y* and the sample rate *FS*, in Hertz. This instruction can read the following audio file types: wave (.wav), FLAC (.flac), MP3 (.mp3), MPEG-4 (.m4a, .mp4), and OGG (.ogg).
- “*soundsc(Y,FS)*,” it writes the *Y* vector into the speaker using the sample rate *FS*.

After the second message “*Reading frame by frame audio file...*” is shown on the screen, the audio message is processed frame-by-frame separately, written to the speaker and sent to the speaker frame-by-frame using a loop instruction. Using the Following MATLAB command from “*Audio Toolbox*” and “*DSP System Toolbox*”:

- “*fileReader = dsp.AudioFileReader(FILENAME)*” allows the specified audio file to be read using an instruction from “*DSP System Toolbox*” that allows reading of audio samples from it and to be stored in the object “*fileReader.*”
- “*deviceWriter = audioDeviceWriter(“SampleRate”, fileReader.SampleRate)*,” it allows the audio data to be played using the computer’s audio device in the object

TABLE 2.5 The MATLAB main program to reproduce “NLP read audio files” as entire audio file and frame-by-frame.

```

%% NLP MATLAB audio file
% Textbook: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% Require: Audio Toolbox and “DSP System Toolbox”
%% Environment
clc; % Clear Command Window
%% Read entire audio file into workspace and then send to speakers
disp("Reading entire audio file...");
[audioData,fs]=audioread("hello_world.wav");
soundsc(audioData,fs);
pause(5);
%% Read each frame of audio into workspace and send it by frame to speakers
disp("Reading frame by frame audio file...");
% Object read audio frame-by-frame.
fileReader = dsp.AudioFileReader("hello_world.wav");
% Object to write audio to speakers
deviceWriter = audioDeviceWriter("SampleRate",fileReader.SampleRate);
% Loop, read each frame from the file and write to the device.
while ~isDone(fileReader)
    i) % Read one frame of audio data from the file.
    ii) audioData = fileReader();
        (1) % Write one frame of audio data to your speakers.
    iii) deviceWriter(audioData);
end
%% Release the file and audio device after use them to free resources
release(fileReader);
release(deviceWriter)

```

Note: This program can be downloaded from the book website companion and install in the following path “...\\Exercises_book_ABME\\CH2\\NLP MATLAB Text to speech\\audiofiles.m”.

TABLE 2.6 The text message shown in the computer screen when the program of Table 2.5 is run in MATLAB.

```

Reading entire audio file...
Reading frame by frame audio file...
>>

```

“deviceWriter” that plays audio samples using an audio output device in real time.

- “While. end,” it is the loop to send each sample to the speakers.

At the end of the program the computer resources are released using the instruction “release (FILENAME).”

2.6.3.1.5 Conclusions and recommendation for research

MATLAB is very easy and practical to use to reproduce different types of audio file as “a complete file” or “frame-by-frame in real-time” in the computer speaker thanks to the functions available in “Audio Toolbox” and “DSP System Toolbox.”

2.6.4 “NLP Text to Speech” using MATLAB

The “NLP Text to Speech capability” is necessary for action generation as explained in Chapter 1, Biomedical Engineering and the Evolution of Artificial Intelligence, in the “General Architecture Framework of a Cognitive Computing Agents System (AI-CCAS),” as shown in Fig. 1.13.

2.6.4.1 Research 2.3: “NLP Text to Speech” as action generation using MATLAB

2.6.4.1.1 General objective

“Create a user input string function for enter text and create text files for “MATLAB NLP Text to Speech function.”

2.6.4.1.2 Specific objectives of this research

1. “Create a user text input string” for “MATLAB NLP Text to Speech.”

2. “Convert Text-to-Speech” from MATLAB from the user input text dialog.

3. “Test Text-to-Speech from different format text file.”

2.6.4.1.3 Procedure

Developing a MATLAB program for this research: “*NLP MATLAB text to speech*” using the code shown in [Table 2.7](#) a. MATLAB has a user function “*TextToSpeech*” that is called from the main program as shown in [Table 2.8](#).

Note*: For this example, it is necessary to install the following MATLAB toolboxes: “*Audio Toolbox*” and “*Text Analytics Toolbox*.”

The function “*TextToSpeech()*” shown in [Table 2.7](#) receives the filename and it extracts the text using the

TABLE 2.7 This MATLAB function “*TextToSpeech()*” is called when the program of [Table 2.8](#) is run to send speech from text.

```
function TextToSpeech(filename)
% Extract text to a string from a file
str=extractFileText(filename);% Instruction from Text Analytics Toolbox
% Make a .NET assembly visble to MATLAB
NET.addAssembly('System.Speech');
% Representation of a MATLAB NET object
obj=System.Speech.Synthesis.SpeechSynthesizer;
obj.Volume=100;
% call function to send the string in object to speaker
Speak(obj, str);
End
```

Note: This program can be downloaded from the website companion and install in the following path “...\\Exercises_book_ABME\\CH2\\NLP MATLAB Text to speech\\TextToSpeech.m”.

TABLE 2.8 This MATLAB program called function “*TextToSpeech()*” used input dialog and different text types to test it.

```
%% NLP MATLAB Text-to-Speech
% Textbook: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% Require: “Text Analytics Toolbox” and “Windows.NET”
%% Environment
clc;% Clear Command Window
%% Test 1) Text-to-Speech from user input dialog
% create a user input dialog
userPrompt='Enter text for speech in the computer speakers?';
titleBar='TextToString';
defaultString='APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS ';
cellInput=inputdlg(userPrompt, titleBar, 2, {defaultString});
if isempty(cellInput)
    a) return;
end;% When clicked Cancel.
str=char(cellInput);% Convert from cell to string.
filename='userInput.txt';
fileID=fopen(filename,'w');fprintf(fileID,str);fclose(fileID);
TextToSpeech(filename);
%% Read Text-to-speech from different format text type file
%% Test 2) Text-to-speech from text file
disp("Text-to-Speech from text file (.txt)...");
TextToSpeech("hello_world_book.txt");
%% Test 3)Text-to-speech from Microsoft Word file
disp("Text-to-Speech from Microsoft Word file (.docx)...");
TextToSpeech("hello world.docx");
%% Test 4)Text-to-speech from HTML pages
disp("Text-to-Speech from html (.htm)...");
TextToSpeech("hello world.htm");
%% Test 5)Text-to-speech from pdf files
disp("Text-to-Speech from pdf file (.pdf)...");
TextToSpeech("hello_world_book.pdf");
```

Note: This program can be downloaded from the book website companion and install in the following path “...\\Exercises_book_ABME\\CH2\\NLP MATLAB Text to speech\\MATLABTextToSpeech.m”.

“extractFileText()” instruction from “Text Analytics Toolbox.” Then using a Windows .NET assembly object representation is used to call the function “Speak()” to send the text as speech to the computer speakers.

2.6.4.1.4 Results from “NLP Text to Speech” with MATLAB

When the program shown in Table 2.8 is run, the “user input dialog” is shown on the screen as indicated in Fig. 2.11A where an enter user text is requested or press the button “OK” to accept the default text, then the “test to speech function” is called to “send the speech to the computer speakers”; after that the four tests shown in Fig. 2.11B from different text files are listened to from the formats: “text file (.txt),” “Microsoft Word file (.docx),” “HTML web format (.htm),” and “pdf (.pdf).”

2.6.4.1.5 Conclusions and recommendation for research: “NLP Text to Speech” with MATLAB

The function created in this example is very important for “NLP processes” and it is going to be frequently used for different “NLP MATLAB applications,” as an important “action of generation” section of the “General Architecture Framework of a Cognitive Computing Agents System (AI-CCAS)”, as explained in Fig. 1.13.

2.6.5 “NLP Speech to Text” with MATLAB and IBM Cloud API

The solution for “NLP Speech to Text” with MATLAB is based on the “Audio Toolbox,” which needs an “internet

active connection” and an “active subscription to a speech-to-text services” in “Cloud Services*” such as “IBM Watson Speech to Text API,” “Google Cloud Speech-to-Text API,” or “Microsoft Azure Speech Services API.”

Notes*: “Cloud Services” is any service made available to users on “demand via the internet from a cloud computing provider’s server” as opposed to being provided from a company’s own on-premises servers. “API” is come from the initials of “Application Program Interface” that define a set of routines, protocols and tools for use or build software applications. “API” specifies how the software component should interact.

To build the “NLP Speech to Text” with MATLAB we are going to undertake five steps, as indicated in Fig. 2.12; these are:

2.6.5.1 Research 2.4: “NLP Text to Speech” as action generation using MATLAB and IBM Cloud API

2.6.5.1.1 General objective

“Create an NP text to speech as action generation for the framework AI-Cognitive Computing Agents System (AI-CCAS) in MATLAB using an IBM Cloud API.”

2.6.5.1.2 Specific objectives of this research

Apply the seven steps to create the “NLP MATLAB Speech” application, as shown in Fig. 2.12.

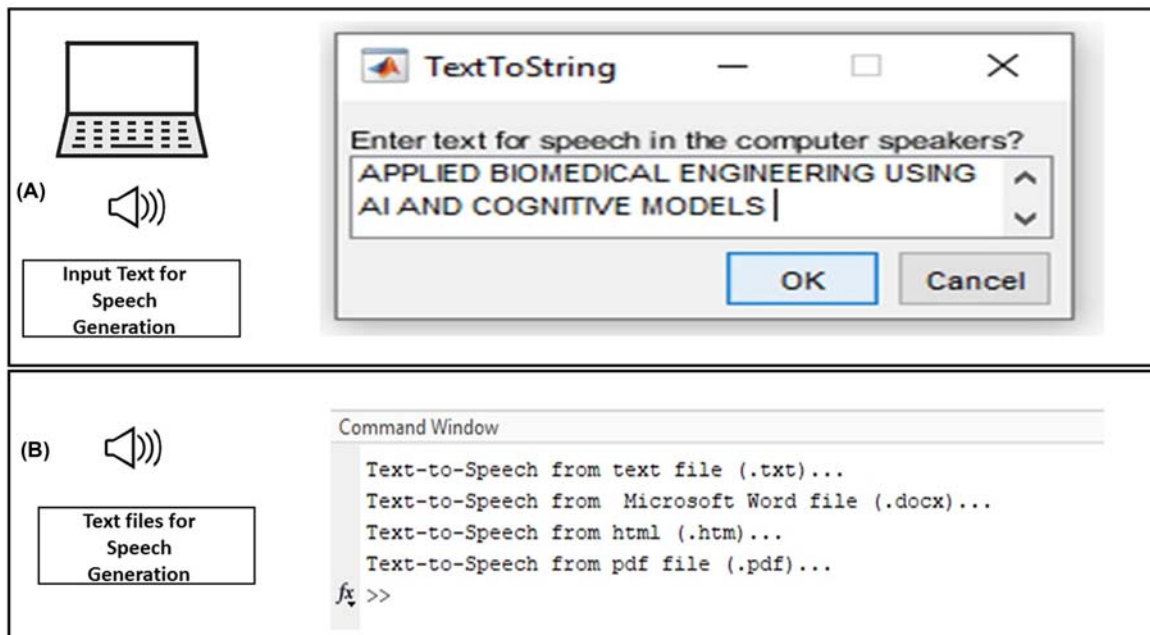


FIGURE 2.11 Output results “NLP Text to Speech” with MATLAB: (A) from user input dialog and (B) from different text files format.

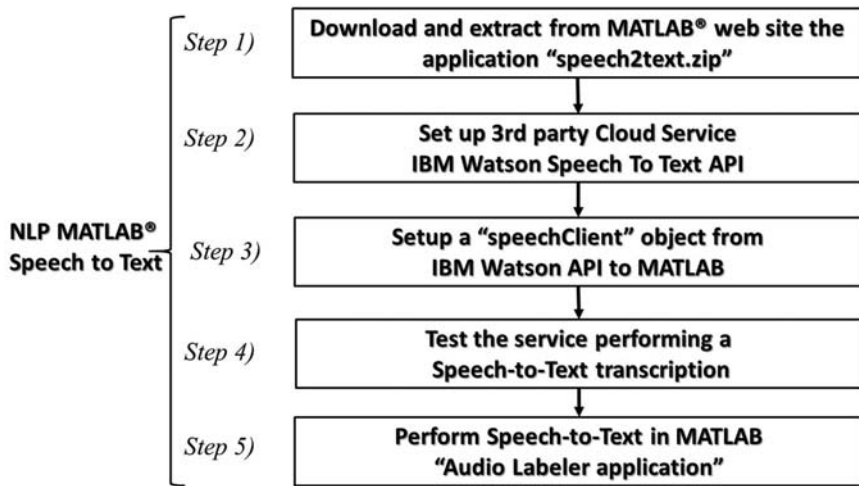


FIGURE 2.12 The five steps to create the “NLP MATLAB Speech” application.



FIGURE 2.13 The first step to create the “NLP MATLAB Speech” application.

2.6.5.1.3 Developing a MATLAB program for this research

Step 1) Download and extract from MATLAB website the application “speech2text.zip” as shown in Fig. 2.13 [30], in the subdirectory for exercises of this book: “..\Exercises_book_ABMECH2\NLP MATLAB Speech to Text.”

This download mainly has three user MATLAB functions: “speech2text.m,” “speechClient.m,” and “Setup.m (inside a subdirectory setup)”, where:

- “speech2text.m” is a function that enables the interface with third party cloud-based speech-to-text APIs
- “speechClient.m” is the specific “speechClient interface with third party cloud-based speech-to-text APIs”

- “Setup.m (inside a subdirectory setup,” run this file in MATLAB to add the “speech2text folder” to the MATLAB search path, and save it for future reference*.

Note*: Please be sure that the MATLAB release is R2019b or newer, and the “Audio Toolbox from MATLAB” must be installed, it contains a “SpeechToText” automation interactive algorithm named “AudioLiber App” to be used later in “Deep Learning” Algorithms.

Step 2) Set up third party Cloud Service IBM Watson Speech to Text API. Create an “IBM Cloud account and its API” following the substep indicated in Table of slides 2.1, showing graphical all the steps to achieve it.

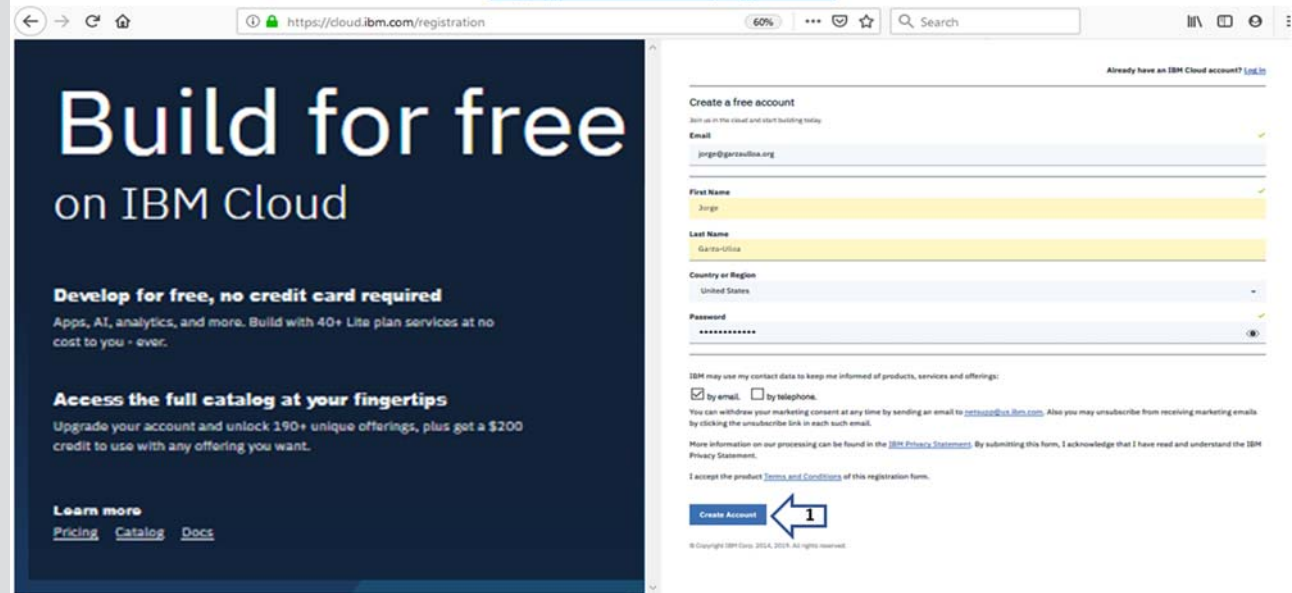
Table of slides 2.1 Steps to set up third party IBM Cloud API key for MATLAB Speech to Text*.

Slide Description

Screen figure

- 1 Login or Create on IBM Cloud Account on the website: <https://cloud.ibm.com/registration>*
Fill the requested fields and press the button: "Next." Then, complete your registration check your email and personal information requested

1. Create an IBM Cloud Account in the website: <https://cloud.ibm.com/registration>

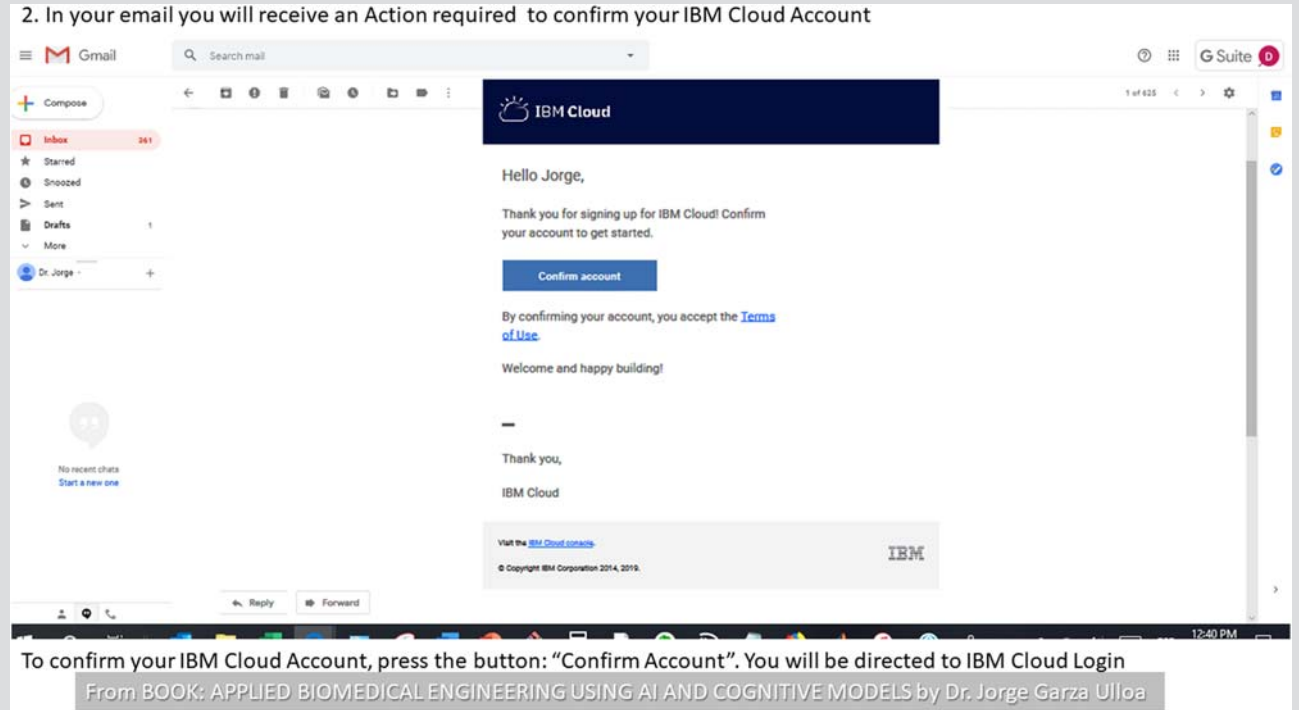


Fill the requested fields and press the button: "Create account". Then, to complete your registration check your email

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 2 In your email you will receive an Action required to confirm your IBM Cloud Account
To confirm your IBM Cloud Account, press the button: "Confirm Account". You will be directed to IBM Cloud Login

2. In your email you will receive an Action required to confirm your IBM Cloud Account

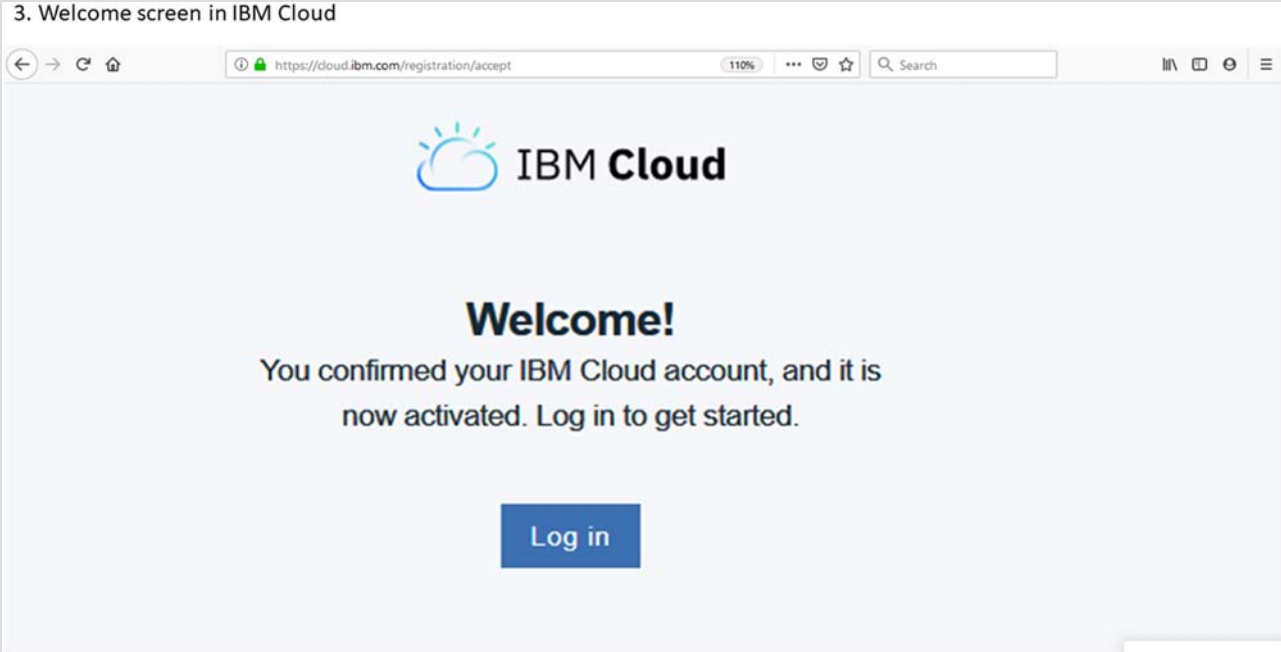


2. In your email you will receive an Action required to confirm your IBM Cloud Account

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

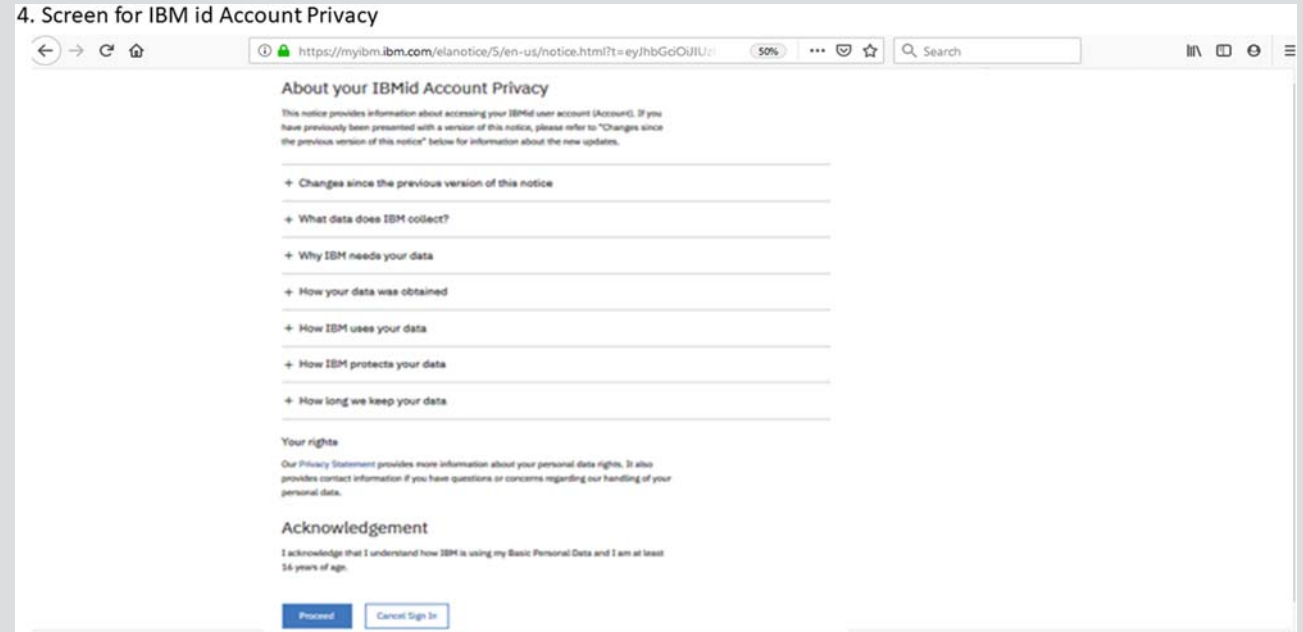
(Continued)

(Continued)

Slide	Description	Screen figure
3	. Welcome screen in IBM Cloud To login press the button: "Log in"	<p data-bbox="669 284 990 305">3. Welcome screen in IBM Cloud</p>  <p data-bbox="669 943 1006 964">To login press the button: "Log in"</p> <p data-bbox="733 976 1815 997">From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

- 4 Screen for IBM id Account Privacy
Read and press the button:
"Proceed"

4. Screen for IBM id Account Privacy



Read and press the button: "Proceed"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

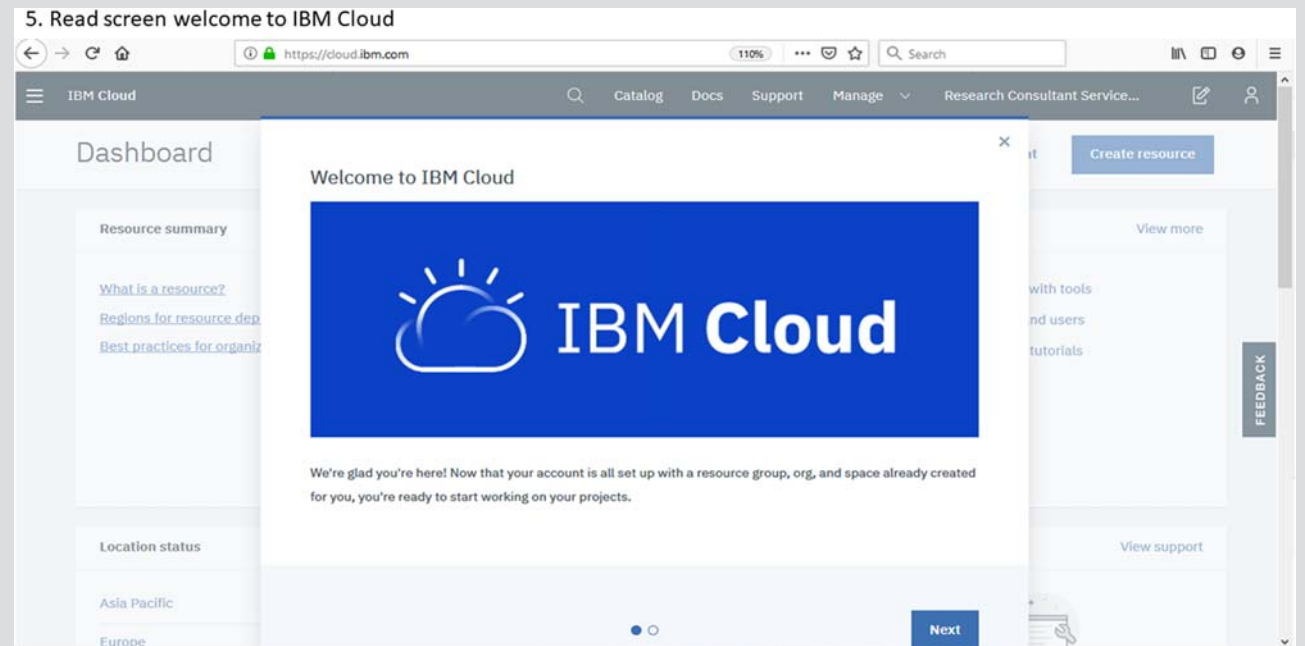
(Continued)

(Continued)

Slide 5 Description
Read screen welcome to IBM Cloud
To continue press the button:
"Next"

Screen figure

5. Read screen welcome to IBM Cloud



The screenshot shows the IBM Cloud dashboard. A modal window titled "Welcome to IBM Cloud" is centered on the screen. The modal features the IBM Cloud logo (a white cloud with a sunburst) on a blue background, followed by the text "IBM Cloud". Below the logo, it says: "We're glad you're here! Now that your account is all set up with a resource group, org, and space already created for you, you're ready to start working on your projects." At the bottom right of the modal is a blue "Next" button. The background dashboard includes a "Dashboard" header, a "Resource summary" section with links like "What is a resource?", "Regions for resource dep", and "Best practices for organiz", and a "Location status" section with "Asia Pacific" and "Europe" options. A "FEEDBACK" button is visible on the right side of the dashboard.

To continue press the button: "Next"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

6 Continue Reading screens about instructions for use IBM Cloud
To continue, until the last screen and press the button: "Close"

6. Continue Reading screens about instructions for use IBM Cloud

Dashboard

What do you want to build?

IBM Cloud

Search Resources

Catalog Docs Support Manage

Research Consultant Service...

Create resource

View more

with tools
and users
tutorials

FEEDBACK

View support

Back Close

We provide a broad set of offerings you can use as building blocks to run your apps. Head over to the catalog and start exploring IBM Cloud.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

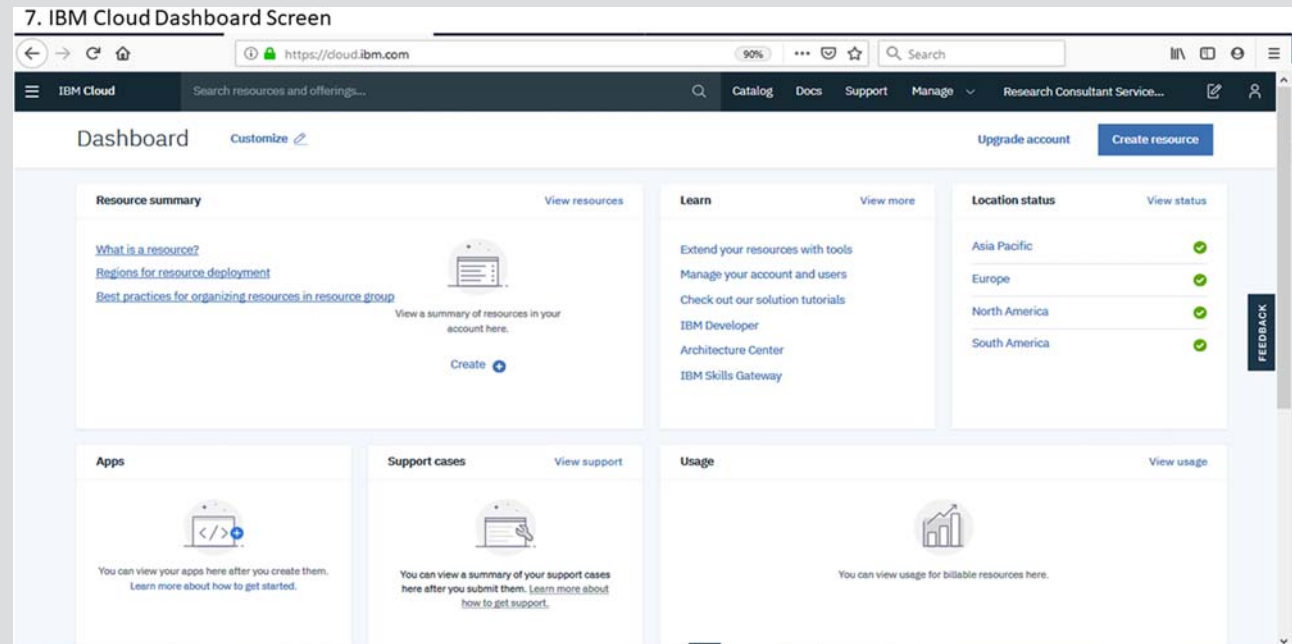
(Continued)

(Continued)

Slide Description

Screen figure

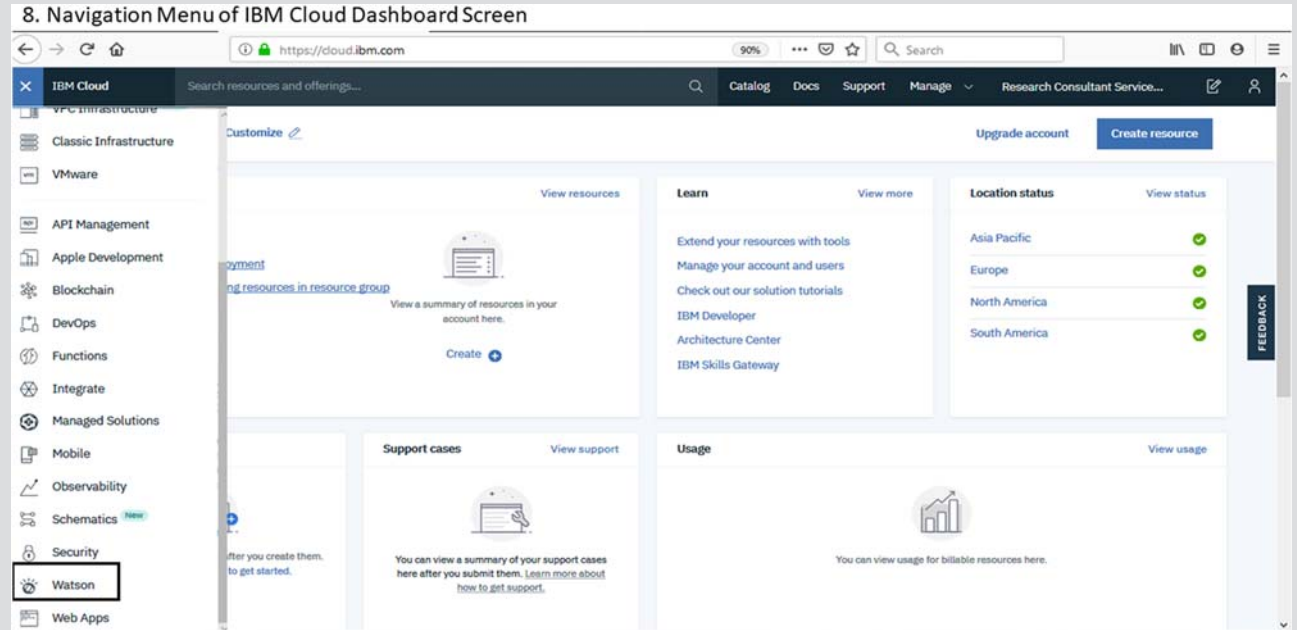
7 IBM Cloud Dashboard Screen
Be familiar with this important screen and select the upper-left icon known as: "Navigation Menu"



Be familiar with this important screen and select the upper-left icon known as: "Navigation Menu"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

8 Navigation Menu of IBM Cloud Dashboard Screen
Go down in the Navigation Menu and locate Watson sub-menu.



Go down in the Navigation Menu and locate Watson

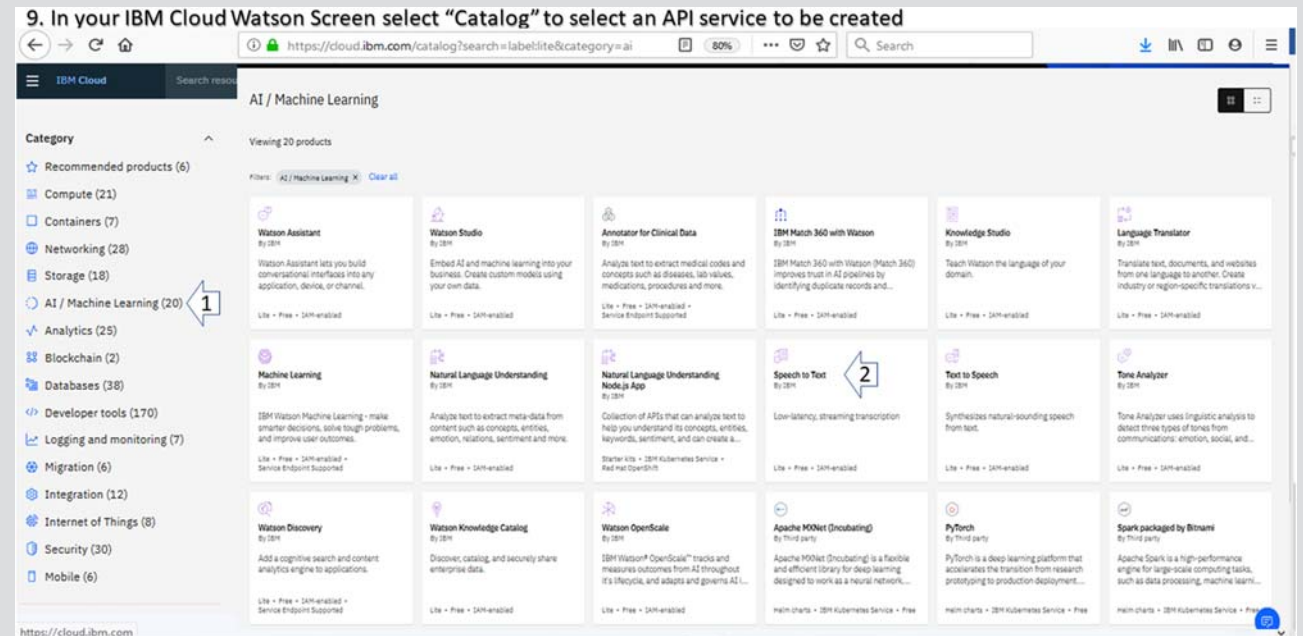
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 9 Description In your IBM Cloud Watson Screen select “Catalog” to select an API service to be created Click on the AI category (arrow1) and select with a click “Speech to Text” (arrow 2)

Screen figure



Click on the AI category (arrow1) and select with a click “Speech to Text” (arrow 2)

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

10 In the IBM Cloud Speech to Text screen

Enter as shown: Service name, accept your suggest Region/ locations, Select group and tag = "nlp," accept license and then press button "Create"

10. In the IBM Cloud Speech to Text screen

The screenshot displays the IBM Cloud console for configuring a Speech to Text resource. The interface includes a navigation bar at the top with 'IBM Cloud' and a search bar. The main content area is titled 'Speech to Text' and 'Lite • IBM'. On the left, there is a sidebar with service details like 'Provider: IBM', 'Updated on: 10/01/2021', and 'Compliance: AI/ Machine Learning'. The main configuration area is divided into several sections: 'Select a location' with a dropdown menu showing 'Dallas (us-south)'; 'Select a pricing plan' with a table of plans; and 'Configure your resource' with input fields for 'Service name', 'Resource group', and 'Tag'. A 'Create' button is located at the bottom right. A 'Summary' sidebar on the right provides a overview of the configuration and a 'Create' button. Red arrows and numbers (1, 2, 3, 4) are overlaid on the image to highlight specific configuration steps.

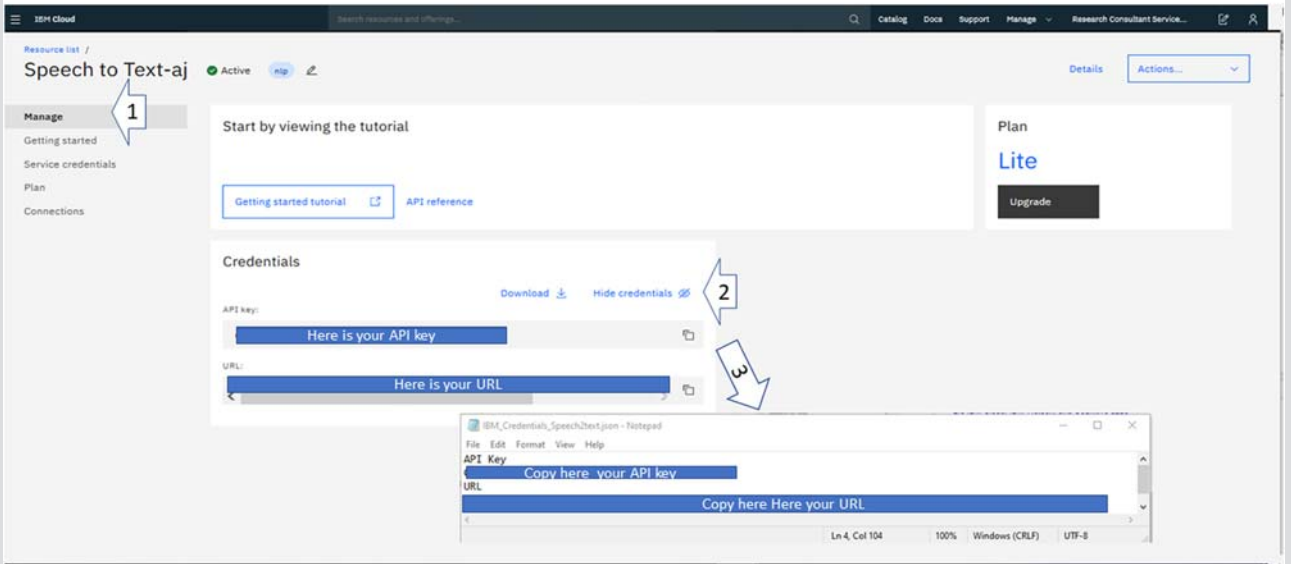
Plan	Features	Pricing
Lite	500 minutes per month The Lite plan gets you started with 500 minutes per month at no cost. When you upgrade to a paid plan, you will get access to Customization capabilities. The plan services are deleted after 30 days of inactivity.	Free
Plus - NEW	Minutes Per Month Simple Volume Price	Click to view tiers and pricing details
Premium - NEW	Everything in Plus Plan, plus... The Premium Plan provides the same features and benefits of using the Plus Plan, but with significantly greater capacity for concurrent transcriptions, enhanced security features to ensure that your data is isolated and encrypted end-to-end while in transit and at rest, and HIPAA readiness. Up to 500 concurrent transcriptions with the option to add more, and 1,000,000 free minutes to start.	

Enter as shown: Service name, accept your suggest Region/locations, Select group and tag="nlp", accept license and press "Create"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide	Description	Screen figure
11	In the IBM Cloud Speech to Text -Resource list/screen Select "Manage" then click "Credentials," then copy them to a new notepad doc, and save the file as "IBM_Credentials_Speech2text.json" in the path "...\Exercises_book_ABME\CH2\nLP MATLAB Speech to Text"	<p data-bbox="671 289 1234 313">11. In the IBM Cloud Speech to Text Resource list / screen</p>  <p data-bbox="671 893 1916 950">Select " Manage" then click Show Credentials to see your assigned API Key and URL, then copy them to a new notepad doc And save the file as "IBM_Credentials_Speech2textAPI.txt" in path "...\Exercises_book_ABME\CH2\nLP MATLAB Speech to Text"</p> <p data-bbox="733 958 1792 982">From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

12 Go back to your IBM Cloud Screen using the Navigator menu and select Dashboard
Notice the service is now on the section “Resource summary,”
logout using the top icon in the right side of screen

12. Go back to your IBM Cloud Screen using the Navigator menu and select Dashboard

The screenshot shows the IBM Cloud dashboard interface. At the top left, the IBM Cloud logo is next to a search bar. A callout '1' points to the search bar. Below the search bar, the 'Dashboard' menu is highlighted, with a callout '2' pointing to it. The main content area features a 'For you' section with several tiles, including 'Build', 'Build cloud-native apps using IBM Cloud Object Storage', 'Build a web app with Watson Speech to Text', 'Explore IBM Cloud Shell', 'Visit the IBM Cloud catalog', 'Create a custom dashboard', 'Use Speech to Text', and 'Backup with Veeam'. Below this is a 'Resource summary' section with a callout '3' pointing to it. The 'Resource summary' section includes a 'Resources' list and a 'Services and software' section. A callout '4' points to a 'Planned maintenance' section. On the right side, there is a 'Research Consultant Services' sidebar with a 'Log out' button, which has a callout '5' pointing to it. At the bottom of the dashboard, there is a footer with the text 'From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa' and a user profile icon.

Notice the service is now on the section “Resource summary”, logout using the top icon in the right side of screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Note*: “IBM Cloud” is a software based on “exponential technologies,” that is constantly being updated and frequently changed, that is, screens, selections, demos, etc. Please, have an “open mind to learn what you need and deduce how to do it in the updated versions.”

Step 3) Set up a “speechClient” object from IBM Watson API to MATLAB

To call the speech client cloud service as API from the “IBM Watson” by a MATLAB@ function, follow the next instructions in the MATLAB screen prompt:

- Run the “*setup.m*” downloaded in Step 1).
- Enter the instruction to create the “*speechObjectIBM*” variable as shown in the top of Table 2.9 for the “*IBM Speech to Text APP*” to create the “*speechClient* object” using the MATLAB instruction indicated at the beginning of the table.

Step 4) Test the service performing a “Speech to Text” transcription

Using the instructions indicated in Table 2.10 “*read a speech signal and get the speech samples (y) and sampling frequency (fs).*” Then, call the “*speech2text*” function and pass the “*speechClientIBM*” object with “*y*” and “*fs*” parameters. The results indicate that the audio transcript = “*several tornadoes touch down as a line of severe thunderstorms swept through Colorado on Sunday*” has a Confidence = 0.96 with Timestamps = 15.

Step 5) Perform Speech-to-Text in MATLAB “Audio Labeler application”

Enter the instruction of MATLAB Audio Toolbox, in the prompt enter “*audio Labeler,*” as shown in Table 2.11; it will call the screen shown in Fig. 2.14A.

TABLE 2.11 This MATLAB instruction opens the audio labeler application.

```
>> audioLabeler
>>
```

Follow the next indications:

- In the “*Audio Labeler*” screen, load the file ‘*audio-file.flac*’ from “*..\Exercises_book_ABME\CH2\NLP MATLAB Speech to Text,*” as indicated in Fig. 2.14A.
- Select the “*Speech to Text*” icon from the Automation section, then select “*Service Name = IBM*” and click on the “*run*” icon as indicated Fig. 2.14B. See the generated text from speech: “*several tornadoes touch down as a line of severe thunderstorms swept through Colorado on Sunday.*”
- To play and listen to the audio press the triangle icon as indicated with an arrow in Fig. 2.14B.
- To label the transaction press the symbol “+” on the section “*File Labels*” then enter the fields as indicated in Fig. 2.15A and press the button “*OK.*”
- Export label definition as an object selection “*Export > Label Definition > To File*” in the current directory with the name “*labelAudioSet.mat*”, as shown in Fig. 2.15B.

TABLE 2.9 This MATLAB instructions creates the “*speechClient* object” for the “*IBM Speech to Text API*”.*

```
>> speechObjectIBM = speechClient('IBM','keywords','example,keywords','keywords_threshold',0.5);
speechObjectIBM.Options
ans =
struct with fields:
keywords: "example,keywords"
keywords_threshold: 0.5000
```

Note*: This program can be downloaded from the website companion and installed in the following path “*..\Exercises_book_ABME\CH2\NLP MATLAB Text to speech\SpeechToTextTranscription.m*”.

TABLE 2.10 This MATLAB instruction perform Speech-to-Text from an audio transcription.*

```
[y,fs] = audioread('audio-file.flac');
tableOut = speech2text(speechObjectIBM,y,fs)
sound(y,fs);
tableOut =
1 × 3 table
Transcript Confidence TimeStamps
"several tornadoes touch down as a line of severe thunderstorms swept through Colorado on Sunday" 0.96 {15 × 1 cell}
```

Note*: This program can be downloaded from the website companion and installed in the following path “*..\Exercises_book_ABME\CH2\NLP MATLAB Text to speech\SpeechToTextTranscription.m*”.

(A) In "Audio Labeler" screen, open file 'audio-file.flac'. (B) Select "Speech to Text" icon from Automation section

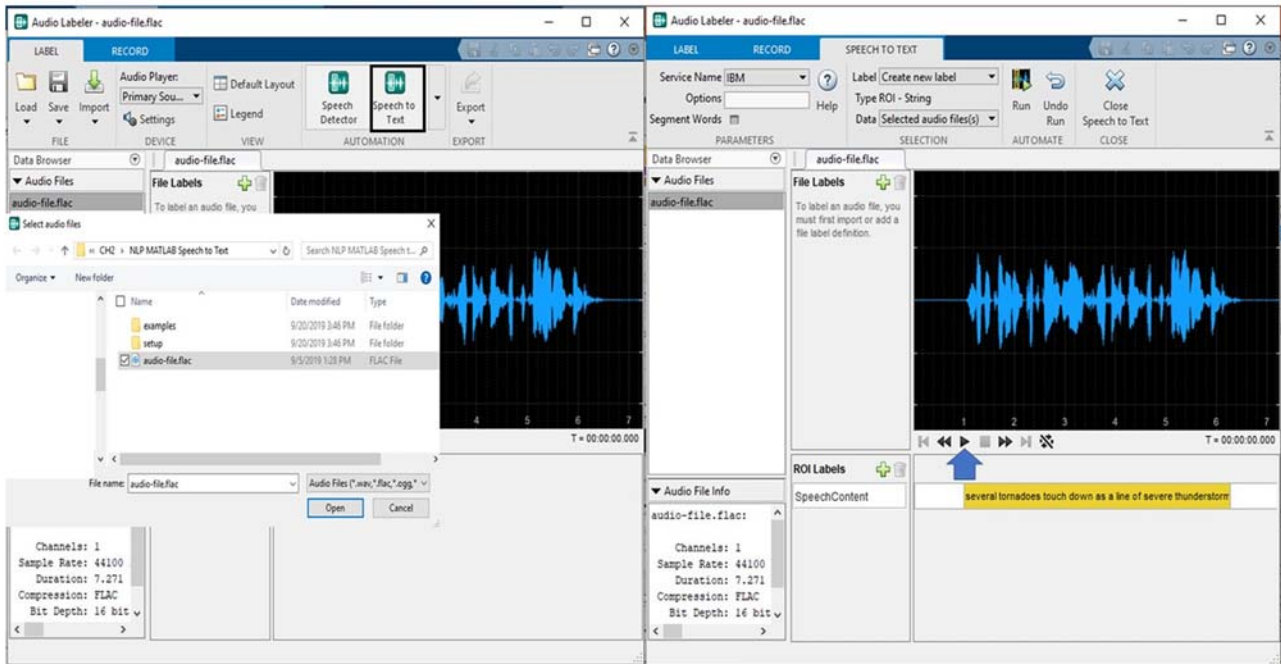


FIGURE 2.14 Perform Speech-to-Text in *audio Labeler* application: (A) opening an audio file. (B) generating the text transcription from the audio file.

(A) In "Audio Labeler" screen, Label the transaction. (B) Export the label definition for ML applications

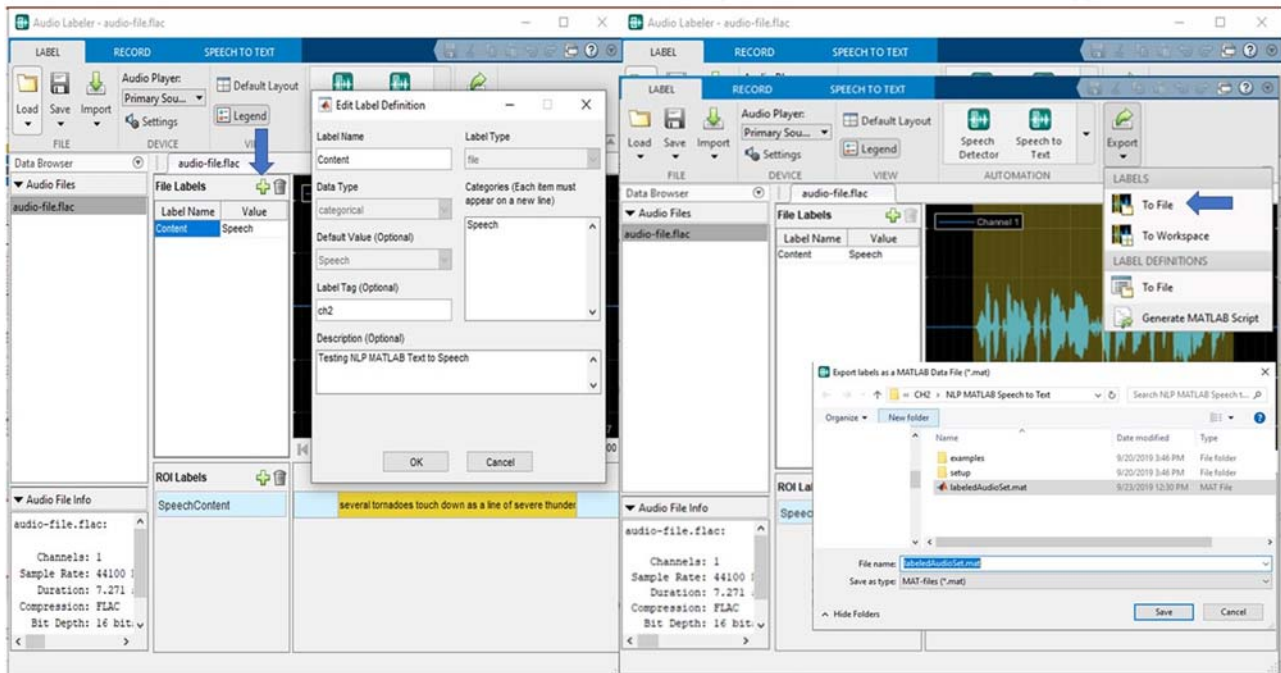


FIGURE 2.15 Performing labeling of a speech to text transaction: (A) label the transaction and (B) export the label to be used in ML applications.

Now the audio file is ready to be used in a ML application as a “*Deep Learning Workflow*,” to be explained in [Chapter 5](#), *Deep Learning Models Applied to Biomedical Engineering*.

2.6.5.1.4 Conclusion and recommendation for this research

All these steps are useful in all your NLP speech to text applications that you will need in this book. In addition, the creation of an IBM account allows the use of many AI Watson applications and API services.

In this section, we show how to set up the MATLAB “*NLP speech to text*” as a step for “*NLP*” making focus on creating a necessary to “*Set up 3rd party API*” due to the big AI resources needed,” in this book we are using “*IBM Cloud services with AI integrated tools in IBM Watson Studio, and others like API services.*”

2.7 Cloud service and AI

A “*Cloud service*” is any service made available to users on demand using the internet from a “*Cloud computing*” provider’s server, instead of the traditional company’s own in-premises servers. “*Cloud services*” are designed to provide easy, scalable access to applications, resources, and services that are fully managed by the “*Cloud service provider*.” A “*Cloud service*” can dynamically scale to meet the need of the users and because the service provider supplies the hardware and software necessary for the request service, there is no need for a company to provision or deploy its own resources or allocate IT staff to manage the cloud service.

2.7.1 Cloud service providers and AI

New technologies such as “*Artificial Intelligence (AI)*,” “*Internet of Things (IoT)*,” “*Big Data analysis*,” “*biotechnology*,” “*digital medicine*,” and many others are currently the focus of many enterprises aiming to simplify their business model, and their processes. Using “*Cloud-based AI solutions*” allows these companies to deploy “*AI apps*” to the average user and simplifies their applications and easy access in the age of “*mobile computing*.”

Based on the kind of services that a Cloud service provider is offering, we can classify them in four ways: “*Infrastructure-as-a-Service (IaaS)*,” “*Platform-as-a-Service (PaaS)*,” “*Software-as-a-Service (SaaS)*,” and “*general cloud services*”:

- “*Infrastructure-as-a-Service (IaaS)* is when the cloud service provider offers the most common cloud

services, such as data storage disks and virtual servers,” that is, *Amazon, Rackspace, Flexiscale, and others.*

- “*Platform-as-a-Service (PaaS)* is when the cloud service provider offers a development platform; that includes operating systems, programming language execution environment, databases, and web servers.” “*PaaS*” has many advantages, such as the operating system can be frequently upgraded and developed, the services can be obtained from diverse sources, and programming can be worked in teams geographically distributed, that is, *Google App Engine, Microsoft Azure, Salesforce, and others.*
- “*Software-as-a-Service (SaaS)* is when the cloud service provider offers access to various software as applications on a pay-per-use basis instead of buying licensed programs,” that is, *Gmail, Google docs, and others.*
- “*Other cloud general services* are when the cloud service provider offers special services such as *integration, security, management, testing as a service, etc.*”

“*Cloud computing* is a term that generally is used to describe data centers that offer to many users over the Internet the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user.” Based on the “*deployment model of cloud computing*,” they can be classified as “*public*,” “*private*,” “*hybrid*,” or “*community clouds*”:

- “*Public* is when whole computing infrastructure is located on the premises of a cloud computing company that offers the cloud services for all public access.”
- “*Private* is when the cloud infrastructure is dedicated solely to one customer/organization.” It is not shared with others, yet it is remotely located. Optionally, the customer/organization have an option of choosing an on-premise private cloud as well, which is more expensive, but they do have a physical control over the infrastructure. The private cloud has more security restrictions with restricted access only to the one customer/organization.
- “*Hybrid* is when the cloud infrastructure includes both private and public clouds, depending on their purpose.” For example, a public cloud can be used to interact with customers, while keeping their data secured through a private cloud.
- “*Community cloud* is when the cloud infrastructure is shared between organizations, usually with shared data and data management concerns.” For example, a “*community cloud*” can belong to a government of a single country. Community clouds can be located both on and off the premises.

There are many “Cloud service providers” that provide “Information Technology (IT)” as a service and AI platforms over the internet. “Cloud computing services” range from full applications and development platforms to servers, storage, and virtual desktops. There are various types of cloud computing services available in the market, such as “IBM Cloud,” “Amazon Web Services,” “Microsoft Azure,” “Google Cloud Platform,” “Salesforce,” “Oracle Cloud,” “Alibaba Cloud,” and many others:

- “IBM cloud” is a full stack cloud platform that offers both: “platform as a service (PaaS) and infrastructure as a service (IaaS) in public, private, and hybrid environments.” With “IBM Cloud IaaS,” organizations can deploy and access virtual IT resources, such as computing power, storage, and networking. It is built with a robust suite of advanced and “AI integrated tools in IBM Watson Studio and other services,” allowing one to start building hundreds of cloud computing services and applications immediately.
- “Amazon Web Services (AWS)” is a subsidiary of “Amazon” that offers reliable, scalable, and inexpensive on-demand cloud computing services and platforms, such as computing power, database storage, content delivery, and other functionality, such as “AI tools, Machine Learning on AWS,” and other services, to help businesses scale and grow for individuals, companies, and governments.
- “Microsoft Azure” is a “cloud computing service” created by Microsoft for building, testing, deploying, and managing applications with “AI tools and Azure ML service and other services” through Microsoft-managed data centers.
- “Salesforce” is a “cloud computing service” that offers services for sales, such as: “Sales Cloud,” “Service Cloud,” “Marketing Cloud,” and “Salesforce Artificial Intelligence integrated with Einstein as an integrated AI Tool” that allows adding of data without special preparation or management models, and many also they offer more cloud services with the goal of predict actions from customers.
- “Oracle Cloud” is a “cloud computing service” offered by “Oracle Corporation” providing servers, storage, network, applications, and services through a global network of Oracle Corporation managed data centers. These services include the “Oracle AI” that are used to build, deploy, integrate, and extend database applications in the cloud.
- “Alibaba Cloud,” also known as “Aliyun,” is a Chinese cloud computing company, a subsidiary of Alibaba Group. “Alibaba Cloud” provides cloud computing services to online businesses and Alibaba’s own e-commerce ecosystem. They include in their

services an “ML platform for AI and ET Brain Cloud’s ultraintelligent AI platform” for solving complex business and social problems.

- There are many others like *MassiveGrid, Rackspace, DigitalOcean, Kamatera, Liquid Web, VMware, Verizon, Navisite, Open Nebula, Pivotal, CloudSigma, Dell Cloud, LimeStone, Quadranet*, etc.

All Cloud service providers have excellent services and they are continuously evolving with regard to AI technologies and new AI applications. In this book, we selected to use “IBM Cloud services using AI integrated tools in IBM Watson Studio and others like API services,” using the free account access that facilitates students and researchers to apply the knowledge, examples, exercises, biomedical projects, and ideas included in this book.

2.8 IBM Cloud, IBM Watson, and Cognitive apps

“IBM Cloud” is a robust suite of advanced data and “AI tools” including “Application Program Interface (API)” that defines a set of routines, protocols, and tools for building software applications, such as the one used in [Section 2.6.5](#), “NLP Speech to Text with MATLAB, with the integration of IBM Cloud APIs,” that specifies how the software component should interact. “IBM Cloud” offers public, private, and hybrid environments. It is built with a robust suite of advanced and “AI-integrated tools in IBM Watson Studio and other services” that allow one to start building hundreds of cloud computing services and applications immediately. The “IBM Cloud” catalog has many products and services that are continuously evolving and growing in number, such as “AI,” “Analytics,” “Databases,” “Storage,” “Integration,” “Web & mobile,” “Web & app,” “Compute,” “IoT,” “Security,” “Network,” and many others.

- “IBM Cloud AI” includes “Watson Studio,” “Watson Machine Learning,” “Watson assistant,” “Watson Discovery,” “Watson Personality Insights,” “Voice Agent with Watson,” “Watson Speech to Text Services,” “Watson Text to Speech,” “Watson Natural Language,” “Watson IoT Platform,” “AI Open Scale Watson Knowledge Catalog,” “Power AI,” and many others.
- “IBM Cloud Analytics” includes “Analytic Engine,” “Apache Spark,” “Db2 Warehouse on cloud,” “SQL Query,” “Streaming Analytics,” and many others.
- “IBM Cloud Databases” includes “Cloudant, IBM Cloud DBs,” “Blockchain,” “Db2 Warehouse,” and many others.

- “IBM Cloud Storage” includes “Block Storage,” “File Storage,” “Object Storage,” “Evault” and many others.
- “IBM Cloud Integration” includes “API connect,” “App Connect,” “Aspera on Cloud,” “Event Streams,” “direct link” and many others.
- “IBM Cloud Web & Mobile” includes *Mobile Foundations*, *Mobile Analytics*, *Push Notification*, *Mapbox*, and many others.
- “IBM Cloud Web & App” includes *Size Up Small Business Intelligence*, *Hazard Hub*, *Risk Engine*, *Health Score*, and others.
- “IBM Cloud Compute” includes “Cloud Virtual servers,” “Mass Storage Servers,” “IBM Cloud Private,” “SAP-Certified Infrastructure” and many others.
- “IBM Cloud IoT” includes “IoT Platform,” “Weather Data APIs,” “Car Diagnostics API,” and many others.
- “IBM Cloud Security” includes “Activity Tracker,” “App ID,” “Network Security,” “SSL Certificates,” “Security Advisor,” and many others.
- “IBM Cloud Network” includes: “Internet Services,” “Virtual Router Appliance,” “DNS,” and many others.

In this book some examples and exercises use “IBM Cloud” and they will focus on services and applications based on: “AI,” “Watson Studio,” “Watson Machine Learning,” “Discovery,” “Natural Language Understanding,” “Visual Recognition,” “APIs,” “Storage,” and others to cover the purpose of this book, which is to analyze biomedical engineering problems, and to obtain AI models to detect, classify, and forecast the process of different illness and injuries of the human body, with a special emphasis on neurologic diseases using AI, machine learning, deep learning and cognitive models.

2.8.1 IBM Cloud solution for natural language processing

The main objective of “NLP is reaching the natural conversation between computer and human being as cognitive computing tool.” As explained before, “NLP” is a subfield of “AI,” “ML,” “DL,” and “CC,” and it is an essential part of the “General Architecture Framework of Cognitive Computing Agents System (AI-CCAS),” as shown in Fig. 1.13, and explained in Section 1.6; NLP covers “speech recognition” as a tool for “natural language understanding,” “speech generation” as a tool for “natural language generation,” “cognitive detection,” “cognitive computing,” and “cognitive model obtainment.”

At this time “IBM Cloud NLP” has many applications as APIs and services and continues to develop more with the objective of simplifying and integrating the many AI solutions that are needed in different fields. Some of these applications are: “Speech to Text,” “Text to Speech,” “Natural Language Understanding,” “Watson

Assistant,” “Compare and Comply,” “Knowledge Catalog,” “Knowledge Studio,” “Language Translator,” “Personality Insights,” “Tone analyzer,” “Voice Agent with Watson,” and many others.

- “Speech to Text” is optimized to process a very high volume of data messages with minimal delay (latency) for streaming transcription.
- “Text to Speech” synthesizes natural-language-sounding speech from text.
- “Natural Language Understanding” analyzes text to extract meta-data from contents such as concepts, entities, emotion, relations, sentiments, and more.
- “Watson Assistant” allows the creation of conversational interfaces into any application, device, or channel.
- “Compare and Comply” allows the processing of governing documents to convert, identify classify, and compare important elements.
- “Knowledge Catalog” allows one to discover, catalog, and securely share enterprises.
- “Knowledge Studio” allows teaching IBM Watson the language of your domain.
- “Language Translator” allows the translation of text, documents, and websites from one language to another, creating industry- or region-specific translations.
- “Personality Insights” derives insights from transactional and social media data using psychology to understand customers’ habits and preferences on an individual level or scale.
- “Tone analyzer” uses linguistic analysis to understand emotions and communication style in text. It can be used to conduct social listening, enhance customer service, and it can be integrated with chatbots.
- “Voice Agent with Watson” allows the creation of a cognitive voice agent that uses Watson services to speak directly with customers using natural language to provide self-service over the phone.

NLP is a key to obtain benefits from the massive unstructured text data that exists in the world, applying AI algorithms to process, organize, and understand all the text that is already available plus the new ones that are generated every day allowing immediate access to related cases that can be applied in medicine and healthcare.

2.8.2 IBM Cloud exercise to create APIs for NLP applications

2.8.2.1 General objective

“Create NLP IBM Cloud APIs and test them using “cURL” or “Git command tool” to integrate them in future applications for Biomedical Engineering research.”

2.8.2.2 Specific objectives of this example

1. “Create an application using the IBM Cloud API for Text to Speech,” applying the assigned: “API Key and URL” to be tested out of the IBM website through “cURL” command-line tools and specify different IBM Watson voices available as shown in Fig. 2.16A.
2. “Test an application for IBM Cloud API for Speech to Text” applying assigned: “API Key and URL” to be tested out of the IBM website through “cURL” command-line tools creating different types of audio files and analyze results based on confidence values.
3. “Create an application IBM Cloud API for Natural Language Understanding” applying assigned: “API Key and URL” to be tested out of the IBM website through “cURL” command-line tools, to analyze the content of a web page for: *sentiment, concepts, categories, entities, and keywords.*

Note: At the end of this exercise, you will have three IBM Cloud APIs and services for “NLP understanding,” “Speech to Text,” and “Text to Speech,” as shown in Fig. 2.16B. These are going to be used in other applications in Biomedical Engineering research that will be proposed in this book.

2.8.2.3 Research 2.5: “Creating more IBM Cloud API services and testing them using command lines with cURL as an open software”

2.8.2.3.1 General objective

“Creating more IBM Cloud API services and testing them using from command lines with cURL as an open software.”

2.8.2.3.2 Specific objectives of this research

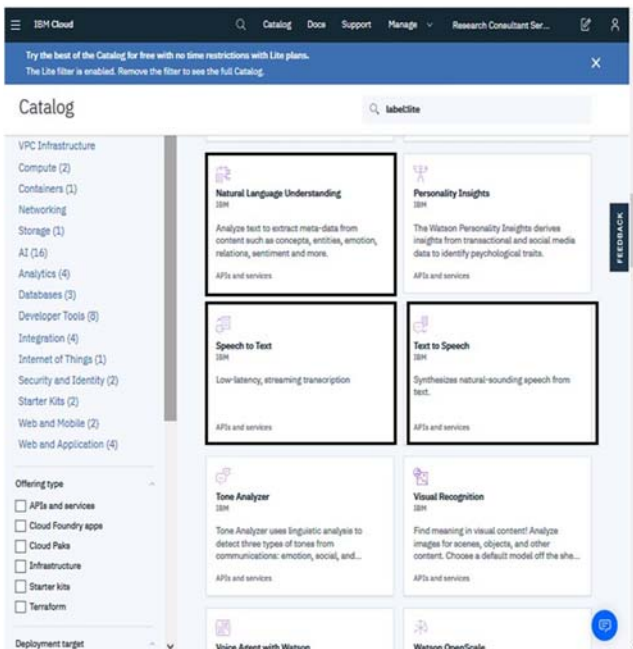
- Create services on IBM Cloud additional to “speech to text”: “text to speech” and “NLP.”
- Learn how to use IBM Cloud API using “IBM credential for API key and URL” from outside languages and command as “cURL” or “GIT.”
- Test in “cURL” the IBM Cloud API service: “text to speech” in different languages and different audio formats.
- Test in “cURL” the IBM Cloud API service: “speech to text” in different languages and different audio formats.
- Test in “cURL” the IBM Cloud API service: “natural language understanding” to analyze text from a web page for sentiment, concepts, categories, entities, and keywords.

2.8.2.3.3 Developing IBM Cloud NLP applications with IBM APIs

The steps descriptions are summarized in Table of slides 2.2, and each step of the example are visually explained using screens sequences.

Note: The IBM Cloud website is evolving in an exponential way every day, some screens could be updated, I recommend to understand very well the objectives, and apply them accordingly with the new screen formats and the contents currently available.

(A) Watson API Services “IBM Cloud™ API NLP Examples”



(B) Resource list for “IBM Cloud™ API NLP Examples”

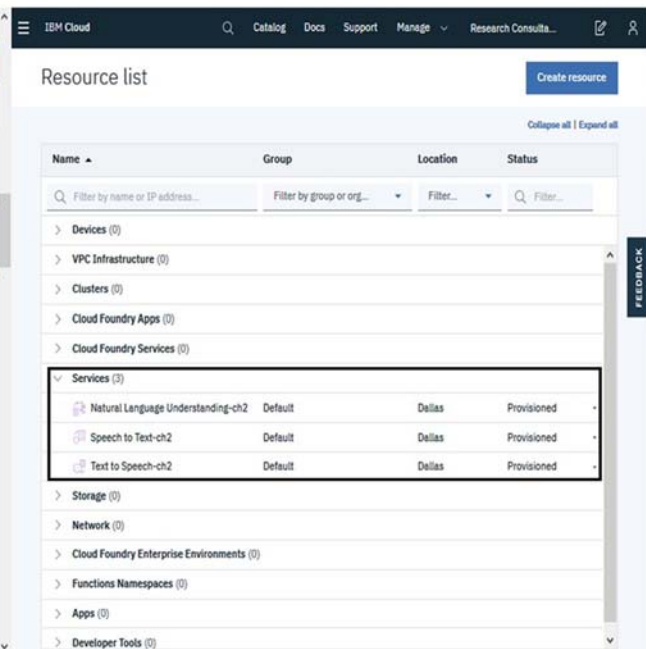
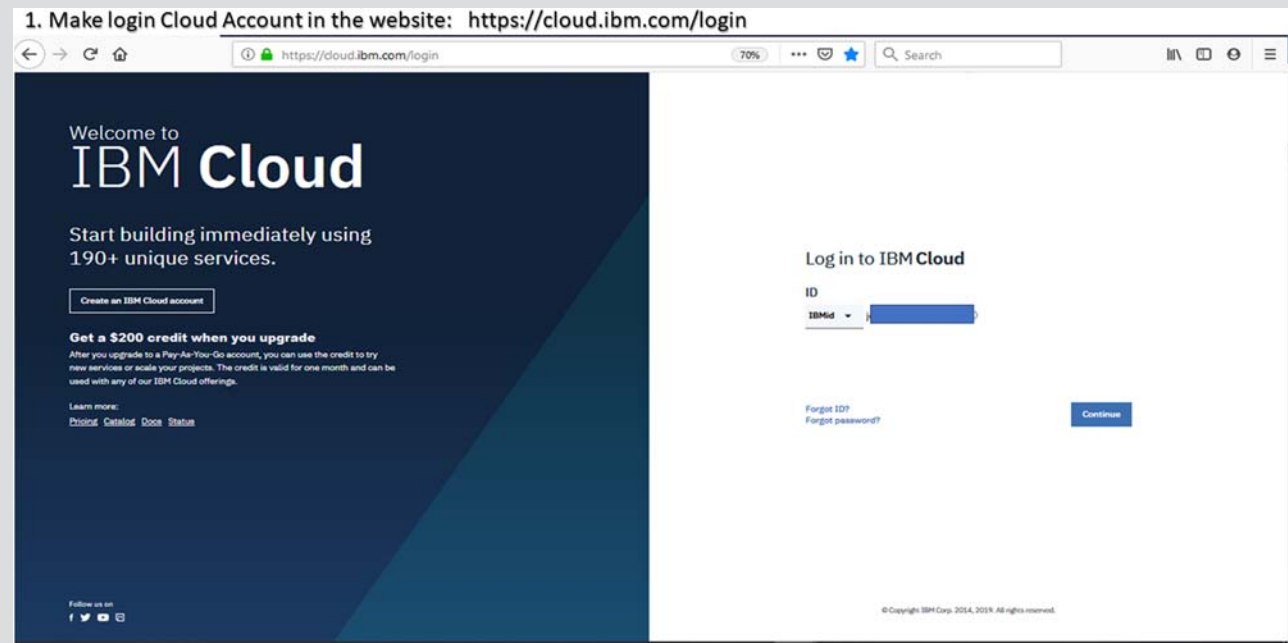


FIGURE 2.16 IBM Cloud Exercise for API NLP: (A) Watson API to create three services and (B) resource list shows the three API services created.

Table of slides 2.2 Steps for the example “IBM Cloud exercise to create APIs NLP”

Slide Description Screen figure

- 1) Login in to the “IBM Cloud” Account in the website: <https://cloud.ibm.com/login>. Enter your “IBMid” obtained in the last research (Example “Ch2–1 IBM Cloud new account” explained in section 2.6.5.1) and press the button “Continue”



Enter your “IBMid” obtained in “Ch2-1 IBM Cloud new account” explain in section 2.6.5.1 and press the button “Continue”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 2) Enter your "IBMid" password and press the button "Log in"

2. Enter your IBMid password

Welcome to
IBM Cloud

Start building immediately using
190+ unique services.

Create an IBM Cloud account

Get a \$200 credit when you upgrade
After you upgrade to a Pay-As-You-Go account, you can use the credit to try new services or scale your projects. The credit is valid for one month and can be used with any of our IBM Cloud offerings.

Learn more:
[Pricing](#) [Catalog](#) [Docs](#) [Status](#)

Follow us on
f t y

Log in to IBM Cloud

ID
IBMid jorge@garzaulioa.org

Password

Forgot ID?
Forgot password?

Log in

© Copyright IBM Corp. 2014, 2019. All rights reserved.

Press the button "Log in"

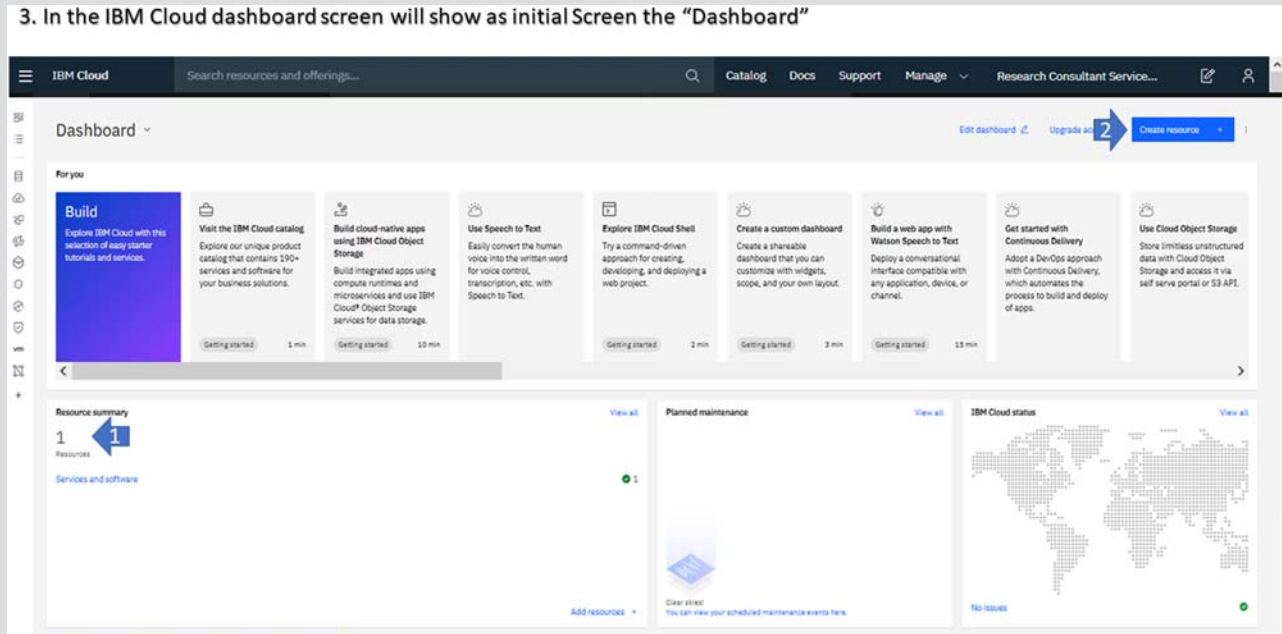
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 3) "IBM Cloud" will show as initial screen the "Dashboard" showing one service for "Speech to Text" created in section 2.6.5.1. Press the button "Create Resource" as indicated.

Screen figure



Showing one service for "Speech To Text" created at section 2.6.5.1, press the button "Create Resource"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 4) In the "IBM Cloud create resources—Catalog" screen
Select the category "AI/
Machine Learning" then
the service "Text to
Speech"

4. In the IBM Cloud create resources – Catalog screen

The screenshot shows the IBM Cloud Catalog interface. The browser address bar displays <https://cloud.ibm.com/catalog?search=labelite&category=ai>. The search bar contains the text "labelite". On the left sidebar, the "Category" list includes "AI / Machine Learning (20)", which is highlighted with a blue arrow labeled "1". The main content area displays a grid of AI services. The "Text to Speech" service is highlighted with a black box and a blue arrow labeled "2". Below the screenshot, the text reads: "Select the category 'AI' then select the service 'Text to Speech'". At the bottom of the screenshot area, a grey box contains the text: "From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa".

Select the category "AI" then select the service "Text to Speech"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

- 5) In the "IBM Cloud Text to Speech" screen
Enter as shown: Service name, select your Region/
Locations, Select group and tag = "nlp," then press
button "Create"

Screen figure

5. In the IBM Cloud Text to Speech screen

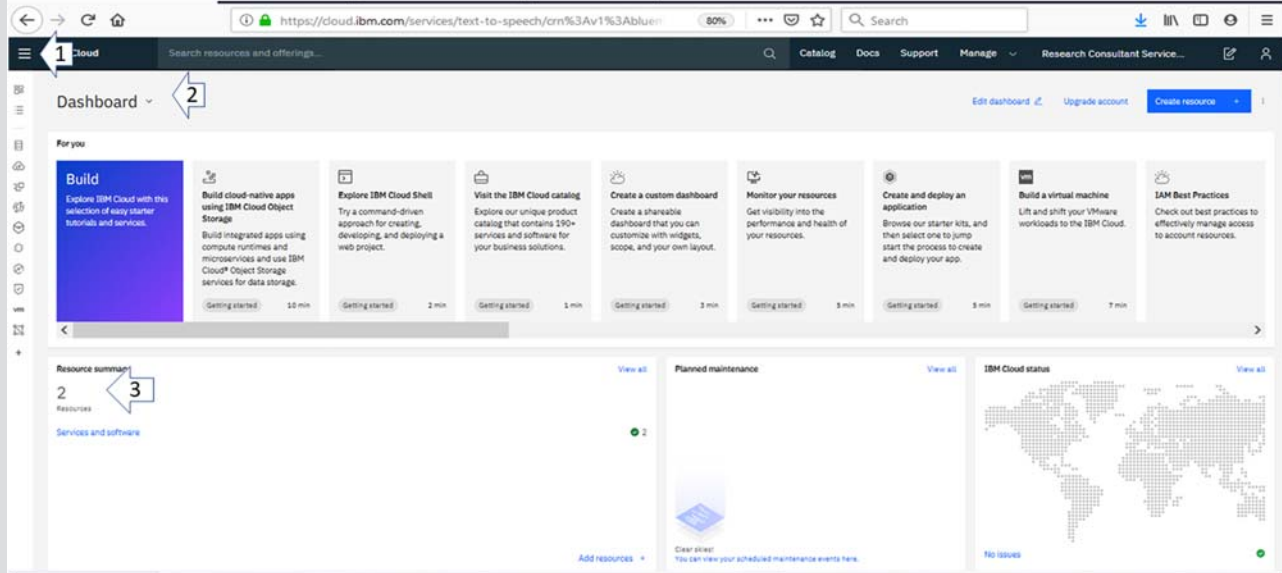
Plan	Features	Pricing
Lite	30,000 Characters per Month The Lite plan gets you started with 30,000 characters per month at no cost. When you upgrade to a paid plan, you will get access to Customization capabilities. Lite plan services are deleted after 30 days of inactivity.	Free
Standard	Standard Characters	\$0.02 USD/THOUSAND CHARS
Premium	Everything in Standard plan... Usage and Training Data is Private • Stored in an Isolated Single Tenant Environment High Availability and Service Level Uptime Guarantee IBM Cloud Service Endpoints HIPAA - Washington DC Only Custom Voice (beta)	

Enter as shown: Service name, select your Region/locations, Select group and tag="nlp", accept license & press "Create"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 6) In the "IBM Cloud Text to Speech Resource list" screen
Select "Manage" then click "show Credentials" to see your assigned "API Key and URL," then copy them to a new notepad doc and save the file as "CH2Text-to-SpeechAPI.txt" in the path "...\Exercises_book_ABME\CH2\NLP MATLAB Speech to Text"

7. In the IBM Cloud select the navigation menu at upper left the option Dashboard



Verify that now you have 2 resources as a services: one for Speech to Text and Test to Speech

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 7) In the IBM Cloud select the navigation menu at upper left the option Dashboard
Verify that now you have 2 resources as a services: one for Speech to Text and Test to Speech

Screen figure

8. For testing the IBM Watson API Install the latest version of :“cURL” or “Git”, they allow commands to call methods service

cURL is a open source computer software project providing a library and command-line tool for transferring data using various protocols.

cURL is used in command lines or scripts to transfer data. It is also used in cars, television sets, routers, printers, audio equipment, mobile phones, tablets, set top boxes, media players and is the internet transfer backbone for thousands of software applications

You can download the latest version at: <https://curl.haxx.se/>

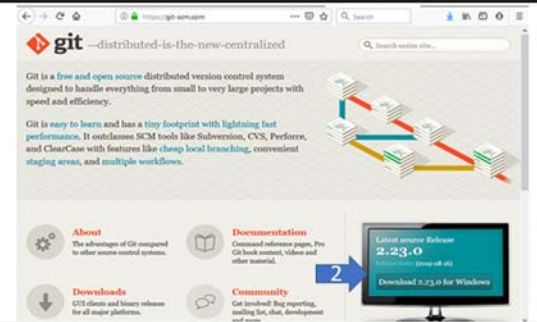


I will recommend to install “Git” instead if you are using windows

Git is a free and open-source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

Git is easy to learn and has a tiny footprint with lightning-fast performance. It outclasses SCM tools like Subversion, CVS, Perforce, and ClearCase with features like cheap local branching, convenient staging areas, and multiple workflows.

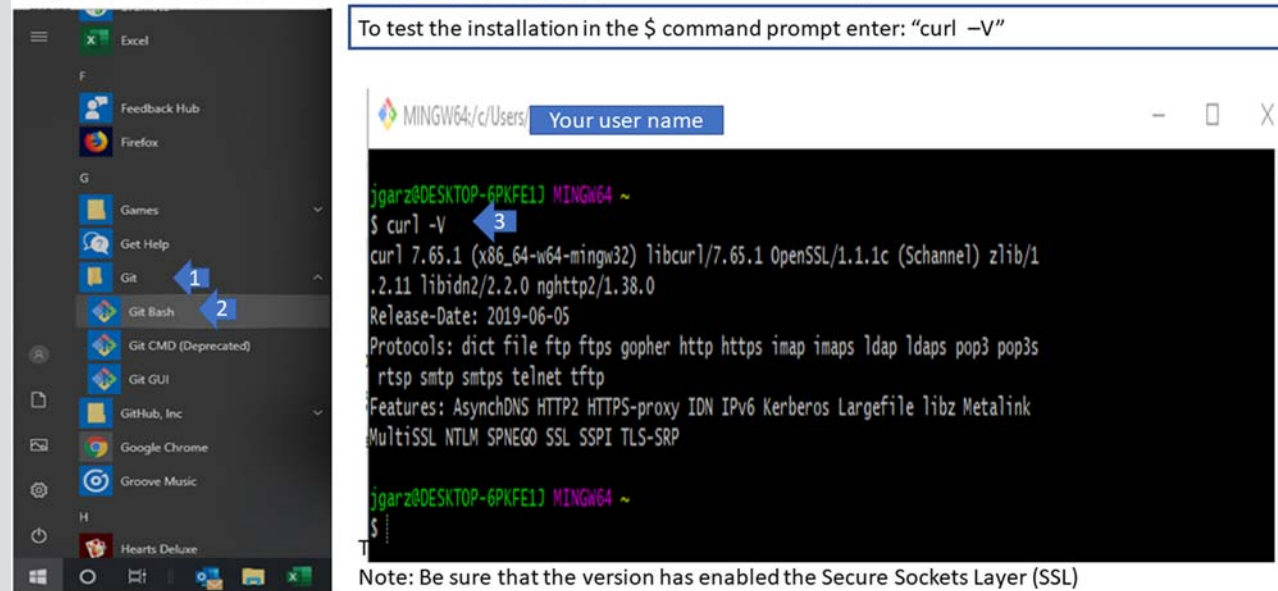
You can download the latest version at: <https://git-scm.com/>



From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 8) For testing the "IBM Watson API" Install the latest version of: "cURL" (link: <https://curl.haxx.se/>) or "Git command tool" (link: <https://git-scm.com>), they allow commands to call methods to test the IBM Watson API service
Note: I will recommend installing "Git command tool" if you are using windows

9. Open the application for "Curl" or "Git" In your computer. *Here is shown the GIT application on windows



On the left side is indicate how to find "Git" installed in windows, in right side of the screen is the "Git" application

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

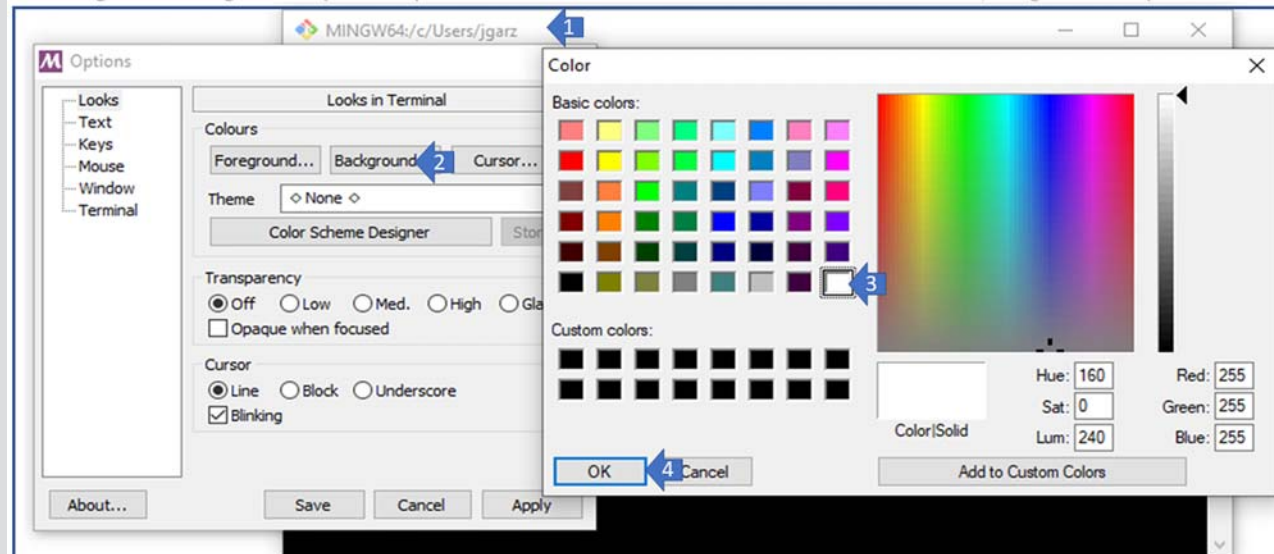
(Continued)

(Continued)

Slide 9) Open the application for "cURL" or "Git command tool" in your computer." *Here is shown the "GIT application on windows." To test the installation in the "\$ command prompt" enter: "curl -V," Note: Be sure that the version has enabled the "Secure Sockets Layer (SSL)"

Screen figure

10. Change "Git" background In your computer from black to white. Place cursor inside "Git" screen, a right click on your mouse



Select "option", then tab for "Background", then on the Color windows select with color as shown. In the same way on the tab for "Foreground", change the color to black. Finally: Press buttons "ok" & "Save"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 10) Change "Git command tool" background in your computer from black to white; for this place cursor inside "Git command tool" screen, a right click on your mouse select "option," then tab for "Background," then on the "Color windows" menu option selects with "color" as shown. In the same way on the tab for "Foreground," change the "color to black." Finally: Press buttons "ok" & "Save"

11. Test your IBM API "Text to Speech" on "Git" synthesizing text in .wav file format in US English

Step 1) Create a sub-directory in the root of your hard drive inside of the username: C:\Users\jgarz\book_ABME (Note: Instead of " ..." will be your user in the windows computer) and Change your default directory in "Git" as indicated:



```
MINGW64:/c:/l ... /jgarz/book_ABME
jgarz@DESKTOP-6PKFE1J MINGW64 ~
$ cd book_ABME
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
$ |
```

Step 2) Copy or enter the following method after you update your assigned IBM Watson API Key and URL indicated in yellow without any empty spaces, they were saved on in the folder ".\Book Applied BME\CH2Text-to-SpeechAPI.txt" Note. The default in the POST .. V1/synthesize method is USA English voice identified as: "en-US_MichaelVoice"

```
curl -X POST -u "apikey: enter here your assigned IBM Watson API Key " \
--header "Content-Type: application/json" \
--header "Accept: audio/wav" \
--data "{\"text\": \"hello world, this is my IBM API Text to Speech for Natural Language Learning\"}" \
--output hello_world.wav \
"https://enter here your assigned IBM Watson URL /v1/synthesize "
```

See in next slide the "Git" response for synthesizing text in US English using method POST. Be sure to use " as vertical not "

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

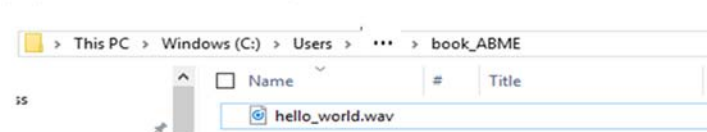
Slide 11) Test your "IBM Cloud API Text to Speech" on "Git command tool" synthesizing text in .wav file format in US English. Create a subdirectory in the root of your hard drive inside of the user name: "C:\Users\...\book_ABME" (Note: Instead of "..." will be your user in the windows computer) and Change your default directory in "Git command prompt." Copy or enter the "cURL method" and update your assigned "IBM Watson API Key and URL". See in the next slide the "Git command prompt" response for synthesizing text in US English using the method "POST". Note: Be sure to use "as vertical not"

Screen figure

12. Test your IBM API "Text to Speech" on "Git" synthesizing text in .wav file format in US English

```
"Git" response for synthesizing text in .wav file format US English using the method POST
MINGW64:/c/Users/j... /book_ABME
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
$ curl -X POST -u "apikey:" \
  > --header "Content-Type: application/json" \
  > --header "Accept: audio/wav" \
  > --data '{"text": "hello world, this is my IBM API Text to Speech for Natural Language Learning"}' \
  > --output hello_world.wav \
  > "/v1/synthesize"
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 232k 0 232k 100 88 359k 136 ---:--:-- ---:--:-- ---:--:-- 359k
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
$
```

Step 3) Go to windows directory indicated as destination in method POST and find the sound file "Hello_word.wav"



Double click the audio file to listen using the default windows media player.
Note: Use your browser or other sound tools to play audio files



12) Test your "IBM Cloud API Text to Speech" on "Git command tool" synthesizing text in .wav file format in US English Go to windows directory indicated as destination in method "POST" and find the sound file "Hello_word.wav," use your browser or other sound tools to play audio file

13. Test your IBM API "Text to Speech" on "Git" synthesizing text in .ogg file default audio format IBM Watson in US English

Step 1) Copy or enter the following method after you update your assigned IBM Watson API Key and URL indicated in yellow without any empty spaces, they were saved on in the folder ".\Book Applied BME\CH2Text-to-SpeechAPI.txt" Note. The default in the POST .. v1/synthesize method is USA English voice identified as: "en-US_MichaelVoice"

```
curl -X POST -u "enter here your assigned IBM Watson API Key" \
--header "Content-Type: application/json" \
--data '{"text": " C:\Users\jgarz\book_ABME \hello world\"}' \
--output hello_world.ogg \
"https://enter here your assigned IBM Watson URL /v1/synthesize"
```

Step 2) Verify the "Git" response for synthesizing text in US English for .ogg file using the method POST

```
MINGW64: c:/Users/jgarz/book_ABME
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
$ curl -X POST -u "apikey:bQ5g9VFwNC6Q6ZNXax55Ta89r7k3qGajqEFCir4r2inz" \
> --header "Content-Type: application/json" \
> --data '{"text": " C:\Users\jgarz\book_ABME\hello world\"}' \
> --output hello_world.ogg \
> "https://stream.watsonplatform.net/text-to-speech/api/v1/synthesize"
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 166 100 118 100 48 85 34 0:00:01 0:00:01 --:--:-- 120
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
$
```

Step 3) Go to the windows directory indicated as destination in the method POST and find the sound file "Hello_word.ogg"

(Continued)

13) Test your IBM API "Text to Speech" on "Git" synthesizing text in .ogg file default audio format IBM Watson in US English As indicated Copy or enter the cURL method after you update your assigned "IBM Watson API Key and URL" indicated in yellow without any empty spaces, they were saved on in the folder "...\\Book Applied BME\\CH2Text-to-SpeechAPI.txt." As indicated. Verify the "Git command prompt" response for "synthesizing text in US English for .ogg file using the method POST" Show the windows directory indicated as destination in the method "POST" and find the sound file "Hello_word.ogg" generated.

Screen figure

14. IBM Watson API has the following many voices available to be used in NLP applications

These are the voices that are available for each language and dialect, including their type and gender. All voices are available as both *Standard voices* and *Neural voices* (To be used on *Multiple Deep Neural Networks*). If you omit the optional voice parameter from a request, the service uses the standard "en-US_MichaelVoice" by default.

Language	Type	Voice	Gender
Brazilian Portuguese	Standard	pt-BR_IsabelaVoice	Female
	Neural	pt-BR_IsabelaV3Voice	Female
Castilian Spanish	Standard	es-ES_EnriqueVoice	Male
	Neural	es-ES_EnriqueV3Voice	Male
	Standard	es-ES_LauraVoice	Female
	Neural	es-ES_LauraV3Voice	Female
French	Standard	fr-FR_DeniseVoice	Female
	Neural	fr-FR_DeniseV3Voice	Female
German	Standard	de-DE_BirgitVoice	Female
	Neural	de-DE_BirgitV3Voice	Female
	Standard	de-DE_DieterVoice	Male
	Neural	de-DE_DieterV3Voice	Male
Italian	Standard	it-IT_FrancescaVoice	Female
	Neural	it-IT_FrancescaV3Voice	Female
Japanese	Standard	ja-JP_EmiVoice	Female
	Neural	ja-JP_EmiV3Voice	Female
Latin American Spanish	Standard	es-LA_SofiaVoice	Female
	Neural	es-LA_SofiaV3Voice	Female
North American Spanish	Standard	es-US_SofiaVoice	Female
	Neural	es-US_SofiaV3Voice	Female
UK English	Standard	en-GB_KateVoice	Female
	Neural	en-GB_KateV3Voice	Female
US English	Standard	en-US_AllisonVoice	Female
	Neural	en-US_AllisonV3Voice	Female
	Standard	en-US_LisaVoice	Female
	Neural	en-US_LisaV3Voice	Female
	Standard	en-US_MichaelVoice	Male
	Neural	en-US_MichaelV3Voice	Male

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 14) "IBM Watson APIs" has the following voices available to be used in "NLP applications: Brazilian Portuguese, Castellan Spanish, French, German, Italian, Japanese, Latin American Spanish, North America Spanish, UK English and US English." If you omit the optional voice parameter from a request, the service uses the standard "en-United States_MichaelVoice voice" by default

15. Use your IBM API "Text to Speech" on "Git" synthesizing text in .wav file format in another language and say your name

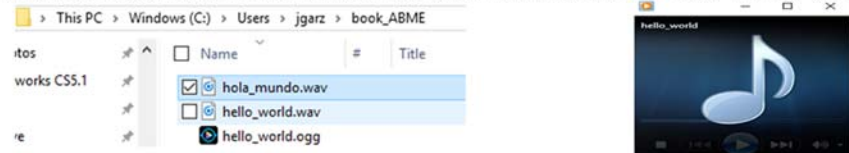
Step 1) Enter the following code for an example on Spanish saying: "Hello world I am jorge"

```
curl -X GET -u "apikey: [redacted]" \
--output hola_mundo.wav \
[redacted]/synthesize?accept=audio%2Fwav&text=hola%20mundo%20soy%20jorge&voice=es-ES-EnriqueVoice"
```

Step 2) Verify the "Git" response for synthesizing text in Spanish using the method POST

```
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
$ curl -X GET -u "apikey: [redacted]" \
> --output hola_mundo.wav \
> "https://api.us-south.text-to-speech.watson.cloud.ibm.com/instances/[redacted]
[redacted]
[redacted]/v1/synthesize?accept=audio%2Fwav&text=hola%20mundo%20
soy%20jorge&voice=es-ES-EnriqueVoice"
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 75084 100 75084 0 0 78786 0 ---:---:---:---: 78704
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
```

Step 3) Go to the windows directory indicated as destination in the method POST and find the sound file "hola mundo" and listen your file



From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 15) Description Use your IBM API “Text to Speech” on “Git” synthesizing text in .wav file format in another language and say your name Enter the following code for an example on Spanish saying: “Hello world I am jorge” . Then verify the “Git” response for synthesizing text in Spanish using the method POST, and finally Go to the windows directory indicated as destination in the method POST and find the sound file “hola_mundo” and listen your file

Screen figure

16. Go back to your IBM Cloud Screen using the Navigator menu and select Dashboard

Notice the two services now on “Resource summary”, we can go with double click verify our assigned “APIkey and url” anytime

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 16) Go back to your "IBM Cloud" Screen using the "Navigator" menu and select "Dashboard" Notice the two services now on "Resource summary," we can go with double click verify our assigned "APIkey and url" anytime

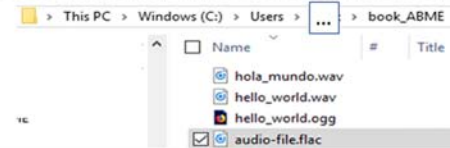
17. Test your IBM API "Speech to Text" on "Git" transcribing US English text in .wav file format without options

Step 1) Change your default directory in "Git" as indicated:



```
MINGW64:/c:/Users/jgarz/book_ABME
jgarz@DESKTOP-6PKFELJ MINGW64 ~
$ cd book_ABME
jgarz@DESKTOP-6PKFELJ MINGW64 ~/book_ABME
$
```

Step 2) Copy the file audio-file.flac from your exercises directory "..\Exercises_book_ABME\CH2 to your working directory ".. C:\Users\...\book_ABME



Step 3) Copy or enter the following method after you update with your assigned IBM Watson API Key and URL indicated in yellow without any empty spaces, they were saved on in the folder "..\Book Applied BME\CH2Speech-to-TextAPI.txt"

```
curl -X POST -u "apikey: enter here your assigned IBM API Key" \
--header "Content-Type: audio/flac" \
--data-binary @audio-file.flac \
" https://enter here your IBM URL for speech to text /v1/recognize"
```

See in the next slide the "Git" response for recognize text in US English using the method POST

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

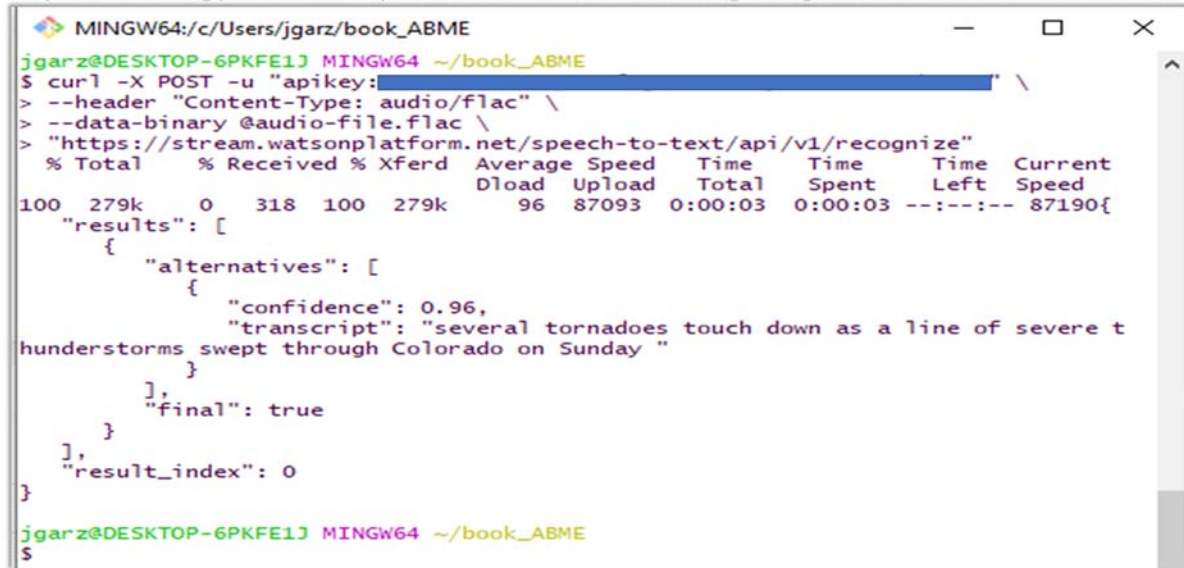
(Continued)

(Continued)

Slide 17) Description
Test your "IBM Cloud API Speech to Text" on "Git command tool" transcribing US English text in .wav file format without options
As indicated in steps 1) to 3) in the screen sequences. See in the next slide the "Git command tool" response for recognize text in US English using the method "POST"

Screen figure

18. Response of testing your IBM API "Speech to Text" on "Git" transcribing US English text in .wav file format without options



```
MINGW64:/c/Users/jgarz/book_ABME
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
$ curl -X POST -u "apikey:[REDACTED]" \
> --header "Content-Type: audio/flac" \
> --data-binary @audio-file.flac \
> "https://stream.watsonplatform.net/speech-to-text/api/v1/recognize"
% Total % Received % Xferd Average Speed Time Time Current
Dload Upload Total Spent Left Speed
100 279k 0 318 100 279k 96 87093 0:00:03 0:00:03 --:--:-- 87190{
  "results": [
    {
      "alternatives": [
        {
          "confidence": 0.96,
          "transcript": "several tornadoes touch down as a line of severe t
hunderstorms swept through Colorado on Sunday "
        }
      ],
      "final": true
    }
  ],
  "result_index": 0
}
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
$
```

Take special attention to the confidence value=.96 and the text transcript "several tornadoes touch down as a line of severe thunderstorms swept through Colorado on Sunday "

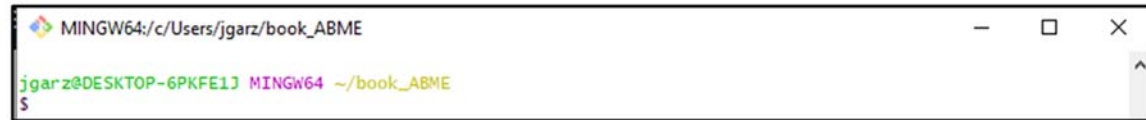
18) Read the response of testing your "IBM API Speech to Text" on "Git command tool" transcribing US English text in .wav file format without options. Take special attention to the "confidence value = 0.96" and the text transcript "several tornadoes touch down as a line of severe thunderstorms swept through Colorado on Sunday"

19. Test your IBM API "Speech to Text" on "Git" transcribing US English text in .flac file format with options

Step 1) Set the parameter

- Set timestamp to true to indicate the beginning and end of each word in the audio stream.
- Set the max_alternatives parameter to 3 to receive the three most likely alternatives for the transcription.
- The example uses the Content-Type header to indicate the type of the audio, audio/flac, and the request uses the default model, en-US_BroadbandModel.

Step 2) Verify that your working default directory in "Git" is "../book_ABME",



```
MINGW64/c/Users/jgarz/book_ABME
jgarz@DESKTOP-6PKFE1J MINGW64 ~/book_ABME
$
```

Step 3) Copy or enter the following method after you update with your assigned IBM Watson API Key and URL indicated in yellow without any empty spaces, they were saved on in the folder "..\Book Applied BME\CH2Speech-to-TextAPI.txt"

```
curl -X POST -u "apikey: enter here your assigned IBM API Key" \
--header "Content-Type: audio/flac" \
--data-binary @audio-file.flac \
"https://enter here your IBM URL speech to text /v1/recognize?timestamps=true&max_alternatives=3"
```

See in the next page the "Git" response for recognize transcribing US English text in .flac file format with options

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

19) Test your "IBM Cloud API Speech to Text" on "Git command prompt" transcribing US English text in .flac file format with options
Following the indicated steps 1) to 3) in the screen sequences

20. "Git" response for recognize transcribing US English text in .flac file format with options using IBM API Speech to Text

```
jjgarz@DESKTOP-6PKFEL1 MINGW64 ~/book_ABME
$ curl -X POST -u "apikey:Lm4U-R8XxxKQYtZ6f8gCXX8tSEfyE1snZTGBC3beo8u" \
> --header "Content-Type: audio/flac" \
> --data-binary @audio-file.flac \
> "https://stream.watsonplatform.net/speech-to-text/api/v1/recognize?timestamps=true&max_alternatives=3"
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 281k 0 2587 100 279k 824 91146 0:00:03 0:00:03 --:--:-- 91971{
"results": [
  {
    "alternatives": [
      {
        "timestamps": [
          [
            "several",
              1.0,
              1.52
            ],
            [
              "tornadoes",
                1.52,
                2.15
              ],
            [
              "touch",
                2.15,
                2.49
              ],
            [
              "down",
                5.6,
                5.73
              ],
            [
              "Sunday",
                5.73,
                6.35
              ]
            ]
          ],
          "confidence": 0.96,

```

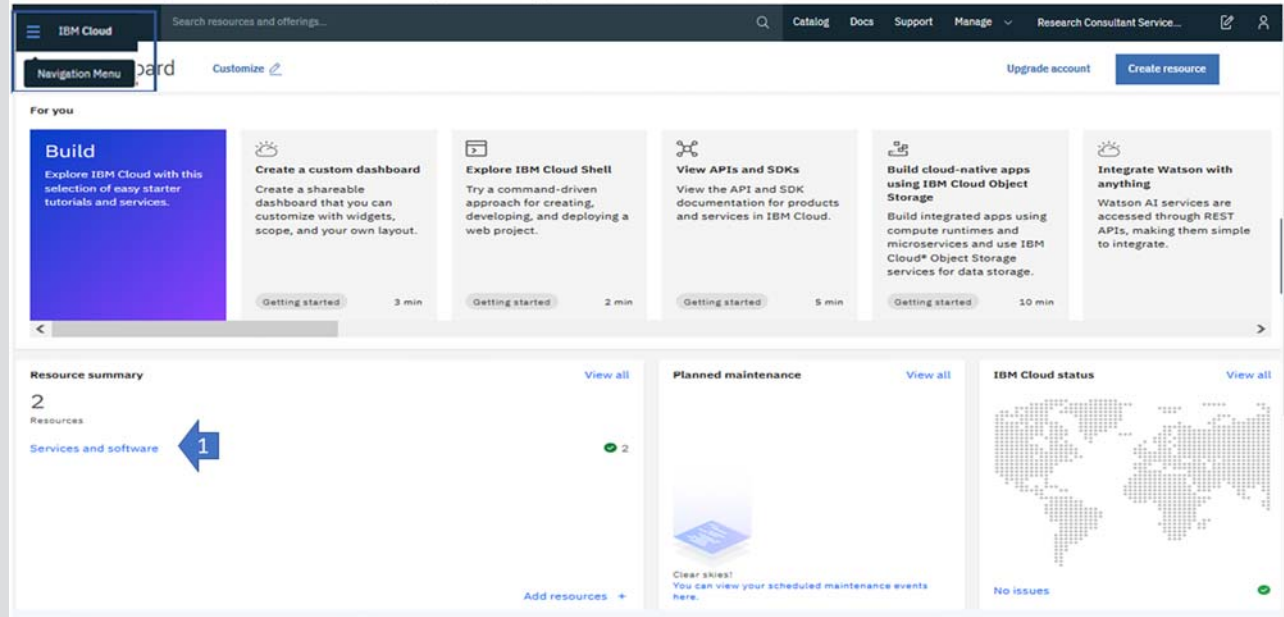
```

            "transcript": "several tornadoes touch down as a line of severe th
understorms swept through Colorado on Sunday "
          },
          {
            "transcript": "several tornadoes touched down as a line of severe t
hunderstorms swept through Colorado on Sunday "
          },
          {
            "transcript": "several tornadoes touch down as a line of severe th
understorms swept through Colorado and Sunday "
          }
        ],
        "final": true
      },
      "result_index": 0
    ]
  }
}
jjgarz@DESKTOP-6PKFEL1 MINGW64 ~/book_ABME
$
```

Pay attention to the timestamp in the left side screen, notice that some words are not shown to simplify the example, and in the right side of the screen the confidence value and three most likely alternatives for the transcription

20) "Git command tool" response for recognize transcribing US English text in .flac file format with options using IBM API Speech to Text Pay attention to the timestamp in the left side screen, notice that some words are not shown to simplify the example, and in the right side of the screen the confidence value and three most likely alternatives for the transcription

21. Go back to your IBM Cloud Screen using the "Navigator menu" and select "Dashboard"



Notice two services created are now on section "Resource summary", click on Services and software to list them

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

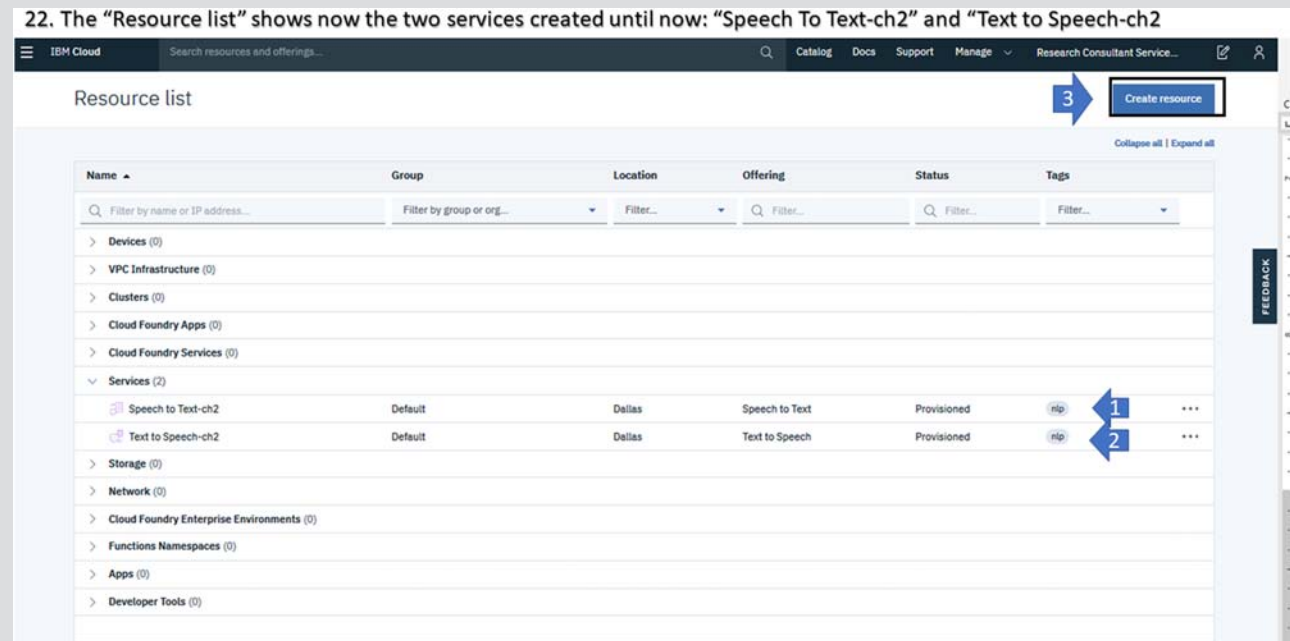
(Continued)

(Continued)

Slide 21) Go back to your "IBM Cloud" Screen using the "Navigator menu" and select "Dashboard" Notice two services created are now on section "Resource summary," double click them to go to the IBM Cloud "Resource List"

Screen figure

22. The "Resource list" shows now the two services created until now: "Speech To Text-ch2" and "Text to Speech-ch2"



Name	Group	Location	Offering	Status	Tags
Filter by name or IP address... Filter by group or org... Filter... Filter... Filter... Filter...					
> Devices (0)					
> VPC Infrastructure (0)					
> Clusters (0)					
> Cloud Foundry Apps (0)					
> Cloud Foundry Services (0)					
Services (2)					
Speech to Text-ch2	Default	Dallas	Speech to Text	Provisioned	nlp
Text to Speech-ch2	Default	Dallas	Text to Speech	Provisioned	nlp
> Storage (0)					
> Network (0)					
> Cloud Foundry Enterprise Environments (0)					
> Functions Namespaces (0)					
> Apps (0)					
> Developer Tools (0)					

Click the button " Create Resource" to create one more for NLP service

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

22) The "Resource list" shows now the two services created until now: "Speech to Text-ch2" and "Text to Speech-ch2" Click the button "Create Resource" to create one more NLP service

23. Create a new service selecting "Natural Language Understanding" in the AI Catalog

The screenshot shows the IBM Cloud AI Catalog interface. At the top, there's a search bar and navigation links for Catalog, Docs, Support, and Manage. A blue banner at the top states: "Try the best of the Catalog for free with no time restrictions with Life plans. The Life filter is enabled. Remove the filter to see the full Catalog." Below this, the main content area is titled "Catalog" and "Viewing 21 products". A search filter "label:lite" is applied, and the results are filtered by "AI / Machine Learning". The left sidebar shows a "Category" list with "AI / Machine Learning (21)" selected, indicated by a blue arrow labeled "1". The main grid displays several service cards, including "Watson Assistant", "Watson Studio", "Annotator for Clinical Data", "IBM Match 360 with Watson", "Knowledge Studio", "Language Translator", "Machine Learning", "Natural Language Classifier", "Natural Language Understanding", "Natural Language Understanding Node.js App", "Speech to Text", and "Text to Speech". A blue arrow labeled "2" points to the "Natural Language Understanding" card. Below the grid, a text overlay reads: "Select AI/Machine Learning and Click the button 'Natural Language Understanding'". To the right of this text, a small card for "Natural Language Understanding" is shown with the text "Lite • IBM • IAM-enabled". At the bottom of the screenshot, a footer reads: "From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa".

(Continued)

(Continued)

- 23) Create a new service selecting "Natural Language Understanding" in the "AI/ Machine Learning Catalog" Click the button "Natural Language Understanding"

Screen figure

24. In the IBM Cloud "Natural Language Understanding" screen

Enter as shown: Select your Region/locations, service name, tag="nlp", select agreement, then press button "Create"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza UJloa

- 24) In the "IBM Cloud Natural Language Understanding" screen
Enter as shown: Select your region/location, service name, tag = "nlp," select agreement then press button "Create"

25. In the IBM Cloud "Natural Language Understanding" screen

Select " Manage" then click show Credentials to see your assigned API Key and URL, then click Download link

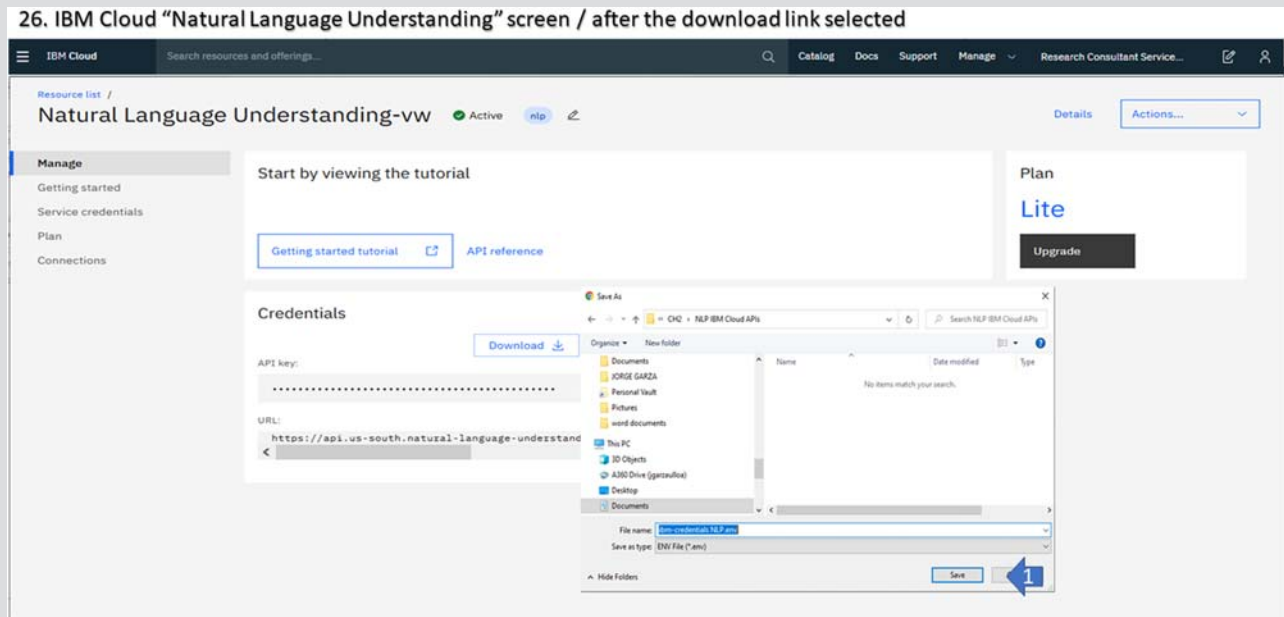
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 25) Description In the IBM Cloud “Natural Language Understanding” screen Select “Manage” then click show Credentials to see your assigned “API Key and URL,” then click “Download link”

Screen figure



Select the directory as “..CH2/NLP IBM Cloud APIs”, with the name as shown “Open with Notepad” and save in a text file in your practice folder as “ibm-credentials.NLP.env”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

26) IBM Cloud "Natural Language Understanding" screen/after the download link selected
 Select the directory as "...CH2/NLP IBM Cloud APIs," with the name as shown "Open with Notepad" and save in a text file in your practice folder as "ibm-credentials NLP.env"

5. In the IBM Cloud Text to Speech screen

Text to Speech
 Synthesize natural-sounding speech from text.

Create | About

Select a location
 Select a location:
 Dallas (us-south) **1**

Select a pricing plan
 Displayed prices do not include tax. Monthly prices shown are for country or location: [United States](#)

Plan	Features	Pricing
Lite	10,000 Characters per Month The Lite plan gets you started with 10,000 characters per month at no cost. When you upgrade to a paid plan, you will get access to Customization capabilities. Lite plan services are deleted after 30 days of inactivity.	Free
Standard	Standard Characters	\$0.02 USD/THOUSAND CHAR
Premium	Everything in Standard plan... usage and Training Data is Private • Stored in an Isolated Single Tenant Environment High Availability and Service Level Uptime Guarantee IBM Cloud Service Endpoints HIPAA - Washington DC Only Custom Voice (Beta)	

Configure your resource

Service name:

Selected a resource group:

Tags: **2**

Access management tags:

Examples: access:dev,proj:version:1

I have read and agree to the following license agreements:
 Terms: 1, 2

Create **4**

Add to estimate

Enter as shown: Service name, accept your suggest Region/locations, Select group and tag="nlp", accept license & press "Create"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 27) Use your "IBM Cloud API Natural Language Understanding" on "Git command prompt" to analyze the next web page: <https://www.elsevier.com/books/applied-biomechanics-using-mathematical-models/garza-ulloa/978-0-12-812594-6>
We need to analyze this web page to get *sentiment, concepts, categories, entities, and keywords*

Screen figure

27. Use your IBM API "Natural Language Understanding" on "Git" to analyze the next web page
<https://www.elsevier.com/books/applied-biomechanics-using-mathematical-models/garza-ulloa/978-0-12-812594-6>

The screenshot shows the Elsevier website product page for the book "Applied Biomechanics Using Mathematical Models" by Jorge Garza Ulloa. The page includes a book cover, title, edition (1st Edition), authors, ISBNs, and pricing options. Three blue arrows point to specific elements: arrow 1 points to the breadcrumb navigation, arrow 2 points to the "Bundle Print & eBook" option, and arrow 3 points to the "DRM-free (Mob, PDF, EPub)" option.

Applied Biomechanics Using Mathematical Models
1st Edition
Authors: Jorge Garza Ulloa
Paperback ISBN: 9780128125946
eBook ISBN: 9780128125953
Imprint: Academic Press
Published Date: 9th June 2018
Page Count: 662

Description
Applied Biomechanics Using Mathematical Models provides an appropriate methodology to detect and measure diseases and injuries relating to human kinematics and kinetics. It features mathematical models that, when applied to engineering principles and techniques in the medical field, can be used in assistive devices that work with bodily signals. The use of data in the kinematics and kinetics analysis of the human body, including

Select country/region:
United States of America
Sales tax will be calculated at check-out

<input checked="" type="radio"/> Bundle Print & eBook	US\$300.00 US\$180.00 40% off
<input type="radio"/> Print - Paperback	US\$150.00 US\$ 112.50 25% off
<input type="radio"/> eBook	US\$150.00 US\$ 112.50 25% off
<input checked="" type="radio"/> DRM-free (Mob, PDF, EPub)	
<input type="radio"/> VitalSource	

[eBook format help](#)

[Add to Cart](#)

We need to analyze this webpage to get *sentiment, concepts, categories, entities, and keywords*
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

28) Test your "IBM Cloud API Natural Language Understanding" on "Git command prompt" analyze a web page to get sentiment, concepts, categories, entities, and keywords
See in the next page the "Git command prompt" the response for "NLP" analyze a web page using the method "POST"

28 . Test your IBM API "Natural Language Understanding" on "Git" analyze a webpage to get sentiment, concepts, categories, entities, and keywords

Step 1) Copy or enter the following "cURL" method in "Git" after you update with your assigned IBM Watson API Key and URL indicated in yellow without any empty spaces, they were saved on folder "..\Book Applied BME\CH2Speech-to-TextAPI.txt

```
curl -X POST -u "apikey: enter here your assigned IBM Watson API Key" \
" https://enter here your assigned IBM Watson URL /v1/analyze?version=2019-07-12" \
--header "Content-Type: application/json" \
--data '{
  "url": "https://www.elsevier.com/books/applied-biomechanics-using-mathematical-models/garza-ulloa/978-0-12-812594-6",
  "features": {
    "sentiment": {},
    "categories": {},
    "concepts": {},
    "entities": {},
    "keywords": {}
  }
}'
```

See in the next page the "Git" response for NLP analyze a webpage using the method POST

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 29) Description
Response from "IBM Closes API Natural Language Understanding" on "Git command prompt" analyze a web page to get: sentiment, concepts, categories, entities, and keywords. Only some results are shown for easy reading, pay special attention to: sentiment, keywords, entities, categories

Screen figure

29 . Response from IBM API "Natural Language Understanding" on "Git" analyze a webpage to get sentiment, concepts, categories, entities, and keywords

```
MINGW64/c/Users/jgarz
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 248 0 0 100 248 0 201 0:00:01 0:00:01 --:--:-- 201{
"usage": {
  "text_units": 1,
  "text_characters": 3000,
  "features": 5
},
"sentiment": {
  "document": {
    "score": 0.765361,
    "label": "positive"
  }
},
"retrieved_url": "https://www.elsevier.com/books/applied-biomechanics-using-mathematical-models/garza-ulloa/978-0-12-812594-6",
"language": "en",
"keywords": [
  {
    "text": "Mathematical Models",
    "relevance": 0.825019,
    "count": 3
  },
  {
    "text": "Dr. Jorge Garza-Ulloa",
    "relevance": 0.68893,
    "count": 1
  }
],
"entities": [
  {
    "type": "Person",
    "text": "Dr. Garza",
    "relevance": 0.958279,
    "count": 3,
    "confidence": 1.0
  },
  {
    "type": "Person",
    "text": "Ulloa",
    "relevance": 0.950192,
    "count": 3,
    "confidence": 0.061792
  }
],
"categories": [
  {
    "score": 0.580418,
    "label": "/education/homework and study tips"
  },
  {
    "score": 0.574215,
    "label": "/technology and computing"
  },
  {
    "score": 0.564981,
    "label": "/health and fitness/disorders"
  }
]
jgarz@DESKTOP-6PKFE1 MINGW64 -
$
```

Only some results are shown for easy reading, pay special attention to: sentiment, keywords, entities, categories

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

30) Go back to your "IBM Cloud Screen" using the "Navigator" menu and select "Dashboard". Notice three service is now on the section "Resource summary," logout using the top icon in the right side of screen

34. Go back to your IBM Cloud Screen using the Navigator menu and select Dashboard

Notice three service is now on the section "Resource summary", logout using the top icon in the right side of screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

2.8.2.3.4 Conclusions

The uses of the “IBM Cloud APIs NLP” are endless because they can be integrated in many other applications, projects, and research through different computer languages and command line tools, such as “cURL” or “Git command tool.”

2.9 The future of the relationship between cognitive science, cognitive computing, and human cognition

In this chapter we have introduced cognitive science, cognitive computing, and human cognitive in relation to helping to obtain solutions to AI biomedical engineering problems. We can conclude that:

- “Understanding consciousness at a cognitive level and looking at its correlation with neural pathways will be a vastly difficult task but will help in the better understanding of how the human brain works and deduct cognitive algorithms for neurological disorders.”
- There are cognitive disorders that are tied to different diseases as explained in many research papers, for example, “A circle and a triangle dancing together: Alteration of social cognition in schizophrenia compared to autism spectrum disorders” [31], “Neural Circuits for Social Cognition: Implications for Autism” [32], “Exploring Relationships Between Negative Cognitions and Anxiety Symptoms in Youth With Autism Spectrum Disorder” [33], and many others.
- “There is not any doubt that the application of cognition and AI will facilitate the modeling of many behaviors studied in cognitive neuroscience, that already are influenced by AI solutions based on reinforcement learning (conditioning tasks) as an area of ML, more useful DL based in ANNs, and many special CC applications, such as NLP and other methods.” All research results will help to explain the association of cognitive and neurological diseases, such as dementia, Parkinson’s disease, Alzheimer’s disease, and many other neurologic disorders [34–36].

In Chapter 7, Cognitive Learning & Reasoning models Models Applied to Biomedical Engineering, we will study many research projects that apply “AI” using “ML-DL-CC” algorithms that can be integrated to the “General Architecture framework of a Cognitive Computing Agents System (AI-CCAS)” with a special emphasis on “cognitive learning” and its relationship with the “neuroscience of reasoning.”

References

- [1] J. Garza-Ulloa, Update on Parkinson’s disease, *Am. J. Biomed. Sci. Res.* 2 (6) (2019). Available from: <https://doi.org/10.34297/AJBSR.2019.02.000614>. AJBSR.MS.ID.000614.
- [2] J. Garza-Ulloa, Paperback ISBN: 9780128125946, eBook ISBN: 9780128125953 June Applied Biomechanics Using Mathematical Models, first ed., Academic Press Elsevier, 2018.
- [3] J. Garza-Ulloa, Chapter 2 - Introduction to human neuromusculoskeletal systems, in: J. Garza-Ulloa (Ed.), Applied Biomechanics using Mathematical Models, Academic Press, 2018, pp. 53–118. ISBN 9780128125946. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00002-0>.
- [4] J. Garza-Ulloa, Chapter 5 - Methods to develop mathematical models: traditional statistical analysis, in: J. Garza-Ulloa (Ed.), Applied Biomechanics Using Mathematical Models, Academic Press, 2018, pp. 239–371. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00005-6>. ISBN 9780128125946.
- [5] J. Garza-Ulloa, Chapter 6 - Application of mathematical models in biomechanics: artificial intelligence and time-frequency analysis, in: J. Garza-Ulloa (Ed.), Applied Biomechanics Using Mathematical Models, Academic Press, 2018, pp. 373–524. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00006-8>. ISBN 9780128125946.
- [6] R. Baillargeon, J. DeVos, Object permanence in young infants: further evidence, *Child Dev.* 62 (1991) 1227–1246.
- [7] H. Beilin, P.B. Pufall, *Piaget’s Theory: Prospects and Possibilities*, Erlbaum, Mahwah, NJ, 1992.
- [8] C. Brainerd, Judgments and explanations as criteria for the presence of cognitive structures, *Psychol. Bull.* 79 (1973) 172–179.
- [9] J.P. Byrnes, Piaget’s Cognitive-Developmental Theory, Reference Module in Neuroscience and Biobehavioral Psychology, Elsevier, 2019. Available from: <https://doi.org/10.1016/B978-0-12-809324-5.23519-0>. <http://www.sciencedirect.com/science/article/pii/B9780128093245235190>. ISBN 9780128093245.
- [10] Available from: <https://nobaproject.com/modules/cognitive-development-in-childhood> (accessed 01.10.19).
- [11] U.C. Goswami, *The Wiley–Blackwell Handbook of Childhood Cognitive Development*, Wiley-Blackwell, 2011.
- [12] Available from: <http://neuromorpho.org/bycell.jsp> (accessed 15.10.19).
- [13] Available from: <https://blogs.scientificamerican.com/brainwaves/know-your-neurons-classifying-the-many-types-of-cells-in-the-neuron-forest/> (accessed 15.10.19).
- [14] W.W. Seeley, et al., Patient-tailored, connectivity-based forecasts of spreading brain atrophy, *Neuron* 104 (5) (2019) 856–868. Available from: <https://doi.org/10.1016/j.neuron.2019.08.037>.
- [15] Available from: <http://library.open.oregonstate.edu/aandp/chapter/14-5-sensory-and-motor-pathways/> (accessed 15.10.19).
- [16] I.H. Bianco, S.W. Wilson, The habenular nuclei: a conserved asymmetric relay station in the vertebrate brain, *Philos. Trans. R. Soc. London. Ser. B Biol. Sciences.* 364 (1519) (2009) 1005–1020. Available from: <https://doi.org/10.1098/rstb.2008.0213>. PMC 2666075. PMID 19064356.
- [17] G. Hernandez, S. Hamdani, H. Rajabi, et al., Prolonged rewarding stimulation of the rat medial forebrain bundle: neurochemical and behavioral consequences, *Behav. Neurosci.* 120 (4) (2006) 888–904. Available from: <https://doi.org/10.1037/0735-7044.120.4.888>. PMID 16893295.

- [18] J.J. Gooley, J. Lu, T.C. Chou, T.E. Scammell, C.B. Saper, Melanopsin in cells of origin of the retinohypothalamic tract, *Nat. Neurosci.* 4 (12) (2001) 1165. Available from: <https://doi.org/10.1038/nn768>. PMID 11713469.
- [19] Available from: <https://www.kenhub.com/en/library/anatomy/neural-pathways> (accessed 09.10.19).
- [20] Available from: <https://neurosciencenews-com.cdn.ampproject.org/c/s/neurosciencenews.com/dementia-brain-networks-15069/amp/> (accessed 15.10.19).
- [21] Available from: https://psychology.wikia.org/wiki/Mesocortical_pathway (accessed 09.10.19).
- [22] R.C. Pierce, V. Kumaresan, The mesolimbic dopamine system: the final common pathway for the reinforcing effect of drugs of abuse? *Neurosci. Biobehav. Rev.* 30 (2006) 215–223.
- [23] T.A. Zhang, R.E. Maldve, R.A. Morrisett, Coincident signaling in mesolimbic structures underlying alcohol reinforcement, *Biochem. Pharmacol.* 72 (2006) 919–927.
- [24] F. Guzmán, n.d. The four dopamine pathways relevant to antipsychotics pharmacology [Video blog post]. <http://psychopharmacologyinstitute.com/antipsychotics-videos/dopamine-pathways-antipsychotics-pharmacology/> (retrieved 15.11.16).
- [25] Available from: <https://www.verywellmind.com/learning-study-guide-2795698> (accessed 20.08.19).
- [26] Available from: <https://explorable.com/biology-of-learning-and-memory> (accessed 20.08.19).
- [27] J.M. Parisi, G.W. Rebok, Q.-L. Xue, et al., The role of education and intellectual activity on cognition, *J. Aging Res.* 2012 (2012) Article ID 416132, 9 pages. Available from: <http://doi.org/10.1155/2012/416132>.
- [28] M.S. Albert, K. Jones, C.R. Savage, et al., Predictors of cognitive change in older persons: MacArthur studies of successful aging, *Psychol. Aging* 10 (4) (1995) 578–589.
- [29] L. Chen, M.-Y. Lee, J. Wu, A nalysis of higher education and management model based on cognitive anthropology, *Cognit. Syst. Res.* 52 (2018) 909–916. Available from: <https://doi.org/10.1016/j.cogsys.2018.08.017>. ISSN 1389-0417.
- [30] MathWorks Audio Toolbox Team, speech2text, MATLAB Central File Exchange, 2019. Available from: <https://www.mathworks.com/matlabcentral/fileexchange/65266-speech2text> (retrieved 19.11.20).
- [31] G. Martinez, E. Mosconi, C. Daban-Huard, M. Parellada, L. Fananas, R. Gaillard, et al., A circle and a triangle dancing together”: alteration of social cognition in schizophrenia compared to autism spectrum disorders, *Schizophrenia Res.* 210 (2019) 94–100. Available from: <https://doi.org/10.1016/j.schres.2019.05.043>. <http://www.sciencedirect.com/science/article/pii/S0920996419302208>. ISSN 0920-9964.
- [32] M. Fernández, I. Mollinedo-Gajate, O. Peñarikano, Neural circuits for social cognition: implications for autism, *Neuroscience* 370 (2018) 148–162. Available from: <https://doi.org/10.1016/j.neuroscience.2017.07.013>. <http://www.sciencedirect.com/science/article/pii/S0306452217304839>. ISSN 0306-4522.
- [33] A. Keefer, N.L. Kreiser, V. Singh, A. Blakeley-Smith, J. Reaven, R.A. Vasa, Exploring relationships between negative cognitions and anxiety symptoms in youth with autism spectrum disorder, *Behav. Ther.* 49 (5) (2018) 730–740. ISSN 0005-7894. Available from: <https://doi.org/10.1016/j.beth.2017.12.002>. <http://www.sciencedirect.com/science/article/pii/S0005789417301314>.
- [34] L.M. Hackel, D.M. Amodio, Computational neuroscience approaches to social cognition, *Curr. Opin. Psychol.* 24 (2018) 92–97. Available from: <https://doi.org/10.1016/j.copsyc.2018.09.001>. <http://www.sciencedirect.com/science/article/pii/S2352250X18300721>. ISSN 2352-250X.
- [35] R.E. Beaty, P. Seli, D.L. Schacter, Network neuroscience of creative cognition: mapping cognitive mechanisms and individual differences in the creative brain, *Curr. Opin. Behav. Sci.* 27 (2019) 22–30. Available from: <https://doi.org/10.1016/j.cobeha.2018.08.013>. <http://www.sciencedirect.com/science/article/pii/S2352154618301219>. ISSN 2352-1546.
- [36] Z. Steine-Hanson, N. Koh, A. Stocco, Refining the common model of cognition through large neuroscience data, *Procedia Computer Sci.* 145 (2018) 813–820. Available from: <https://doi.org/10.1016/j.procs.2018.11.026>. <http://www.sciencedirect.com/science/article/pii/S1877050918323123>. ISSN 1877-0509.

Further reading

- Available from: <https://www.ibm.com/watson-health>.
- Available from: <https://www.ibm.com/watson-health/about>.
- Available from: <https://www.ibm.com/watson-health/life-sciences>.
- Available from: <https://www.ibm.com/watson-health/learn/truven-health-analytics>.
- Available from: <https://www.mathworks.com/help/thingspeak/matlab-toolbox-access.html>.
- Available from: <https://www.mathworks.com/academia/books/matlab-deep-learning-kim.html>.

This page intentionally left blank

Artificial Intelligence Models Applied to Biomedical Engineering

3.1 Introduction artificial intelligence and biomedical engineering

“Artificial intelligence (AI) in Biomedical Engineering (BME)” is the use of smart device software to analyze complex data, classify, model, optimize, and predict many fields of medicine and healthcare. Specifically:

- “AI is the ability for smart computer algorithms to approximate results for the suggestion of useful conclusions.”
- “AI is a computer science branch that uses algorithms, heuristics, pattern recognition, rules, different types of learning, etc. based on Mathematics, Statistics, Data Mining, and others.”
- “AI has the potential to be applied in any almost every field of medicine and healthcare, such as drug development, patient monitoring, personalized patient treatment plans, etc.”
- “AI is patterned after the brain’s neural networks.” It uses multiple layers of nonlinear processing units to “teach itself” how to understand data, classifying the records, or making predictions.

Some “BME applications” where “AI” is currently used are:

- “AI analysis for blood cell disorders”: blood cell disorders are a condition in which there are problems with “red blood cells (RBC),” “white blood cells (WBC),” or the smaller circulating cells called “platelets,” which are critical for clot formation [1].
- “Breast cancer AI analysis”: “breast cancer” is the leading cause of cancer deaths in women today and it is the most common type of cancer in women. The chance that a woman will die from “breast cancer” is about 1 in 38. “Breast Cancer Early Detection and Diagnosis” can be undertaken with various different tests: Mammograms, Breast Ultrasound, Breast MRI,

Newer and Experimental Breast Imaging, Biopsy, and other ways. AI helps in the detection, analysis, and suggestions for treatment.

- “Brain tumor AI detection”: a brain tumor is a mass or growth of abnormal cells in the brain. They can be “benign or noncancerous,” and “malignant or cancerous.” Brain tumors can begin as primary brain tumors in the brain, or secondary metastatic cancer that can begin in other parts of the body and spread to the brain. There are many types of brain tumors with different treatments, such as *Acoustic neuroma, Astrocytoma, Brain metastases, Choroid plexus carcinoma, Craniopharyngioma, Embryonal tumors, Ependymoma, and others.* AI helps in the detection, analysis, and suggestions for specifying treatment for each kind of “brain tumor” [2].
- “Synthesize Electronic Health Record (HER)”: AI can read and understand unstructured data, and has the ability to process natural language, allowing it to read clinical text from any source and identify, categorize, and code medical and social concepts.
- “Insights from patients’ data”: AI can identify problems contained in patients’ historical medical records and summarizes the history of their care around those problems and provides a “cognitive summary of a patient records.”
- “Identify Patients similarities”: AI can identify a measure of clinical similarity between patients, allowing researchers to create dynamic patient cohorts and deduce specific suggestions for the best care path for a given group of patients.
- “Create Medical insight”: AI can read through a complete set of medical literature, such as Medline, and identify the documents that are semantically related to any combination of medical concepts.

There are many other examples where “AI is used on Biomedical Engineering.”

“AI” is explained in this book following the “Artificial Intelligence Evolution” as explained in Chapter 1, Biomedical Engineering and the Evolution of Artificial Intelligence, and summarized in Fig. 1.3 as well as in the following chapters:

- Chapter 3, *Artificial Intelligence models applied to Biomedical Engineering*
- Chapter 4, *Machine Learning Models applied to Biomedical Engineering*
- Chapter 5, *Deep Learning Models Principles Applied to Biomedical Engineering*
- Chapter 6, *Deep Learning Models Evolution Applied to Biomedical Engineering*
- Chapter 7, *Cognitive Learning & Reasoning Models Applied to Biomedical Engineering*

3.2 AI optimization in biomedical engineering

“AI Optimization in BME” refers to finding parameters for maximizing or minimizing objectives while satisfying the constraints of defined functions relative to some dataset of biomedical data. “AI Optimization in BME” results allow the comparison of different choices for determining which one might be the best. In the last decade the development in fields of medicine, healthcare, biomedicine, bioinformatics, etc. have led to an exponential increase in the volume of data that need to be optimized as a preliminary step to finding the most informative and discriminative features that optimally represent the data under research [3].

“Optimization in BME” can be used for:

- “AI Optimization in BME for AI algorithms”: Many “AI algorithms” are developed for the treatment planning in radiation therapy which employ techniques such as multiobjective optimization [4].
- “AI Optimization in BME for identify genes expression signatures”: analysis of a large and heterogenous gene expression data to identify specific groups of genes under experimental conditions, with the objective to understand the biological events associated with different physiological states and identify their genes expression signatures [5].
- “AI Optimization in BME for classification applying segmentation method”: It is used in classification to apply “segmentation,” which is the process of partitioning a digital image into different sections in order to change the representation of the image. This new representation may involve certain characteristics in the image such as curves, edges, color, intensity, or texture. “Segmented images are usually used to determine brain abnormalities using image classification” [6].

- “AI Optimization in BME for Data Mining in Bioinformatics and Medical Informatics: for knowledge discovery”: several problems in different areas of “Data Mining” for “Bioinformatics” and “Medical informatics” such as “Knowledge Discovery” can be viewed as finding the optimal covering of a finite set [7].
- “AI Optimization in BME for Data Mining in Bioinformatics and Medical Informatics for: finding a set of homologs sequences”: in “Bioinformatics” and “Medical Informatics” problems that consist of finding a set of homolog (high similarity) sequences of known function to a given amino acid sequence of unknown function from the various annotated sequence databases [7].
- And many other AI applications for “Optimization in BME.”

“AI Optimization in BME” is used in many AI algorithms from “Machine Learning,” “Deep Learning,” and “Cognitive Computing.” Example of each are:

- “Machine Learning”:
 - “Artificial plant optimization algorithm to detect heart rate and presence of heart disease using machine learning” [8].
 - “Video coding optimizations; machine learning-based video coding optimizations” [9].
 - “Hyperparameter optimization for machine learning models based on Bayesian optimization” [10].
- “Deep Learning”:
 - “Analysis of brain subregions using optimization techniques and deep learning methods in Alzheimer disease” [11].
 - “Biomedical sensors”: such as the “Internet of Things (IoT) medical tooth sensor” for monitoring teeth and food level using bacterial optimization along with “adaptive deep learning neural network” [12], and the “Intelligent sensor for Skin cancer diagnosis” using improved particle swarm optimization and deep learning models [13].
- “Cognitive computing”:
 - “Artificial search agents with cognitive intelligence “: artificial search agents with cognitive intelligence for binary optimization problems, computers and industrial engineering [14].
 - “Optimizing the prediction of cognitive outcome in neurological diseases”: optimized machine learning methods for the prediction of cognitive outcome in Parkinson’s disease [15].
 - “Intelligence technologies to optimize performance in augmenting cognitive capacity”: intelligent technologies to optimize performance: augmenting cognitive capacity and supporting self-regulation of critical thinking skills in decision-making [16].

From the mathematical point of view a “*General Optimization*” is a technique where the objective is to find a vector of optimal design parameters “ x ,” applying the minimization or maximization of some characteristic function “ $f(x)$ ” satisfying constraints in the form of equality or inequality and/or parameters bounds. The “*General Optimization*” formula is shown in Eq. (3.1).

$$\text{General Optimization problem } \min_x f(x) \text{ or } \max_x f(x) \quad (3.1)$$

where: x are the design parameters $x = \{x_1, x_2, \dots, x_n\}$;
equality constraints $G_i(x) = 0$ where: $i = 1, \dots, m_e$;
inequality constraints $G_i(x) \leq 0$ where: $i = m_e + 1, \dots, m$;
parameters bounds x_{lb}, x_{ub} .

The accurate solution in “*General optimization*” depends on the characteristics of the “*fitness function, number of constraints, and the size of the problem to resolve.*” There are three kinds of problems for “*General optimization,*” these are: “*Linear Programming,*” “*Quadratic Programming*” and “*Nonlinear Programming*”:

1. “*Linear Programming (LP)*” when both the objective function and the constraints are linear functions of the design variable, for this case reliable solution procedures are readily available.
2. “*Quadratic Programming (QP)*” when the minimization or maximization of a quadratic objective function is linearly constrained, for this case also reliable solution procedures are readily available.
3. “*Nonlinear Programming (NP)*” where the objective function and constraints can be nonlinear functions of the design variables. For this case there are not reliable solution procedures.

AI optimization in Biomedical Engineering is frequently used in many subfields applying “*Evolutionary Algorithms as General optimization*” that depend on the characteristics of the “*fitness function, number of constraints, and the size of the problem to resolve,*” as studied in the next section.

3.3 Evolutionary algorithms for AI optimization in BME

“*Evolutionary algorithms*” are a subset of “*evolutionary computation.*” They are based on the concept of “*Artificial Evolution algorithms*” developed using the “*evolution of the species,*” trying to emulate natural evolution as explained by the main “*evolutionary theory of Charles Darwin*” [17]. This is a recommended method for “*AI optimization in BME.*”

“*Evolutionary algorithms*” are represented by the existence of a population of individuals known as “*genes*”

exposed to environmental pressure or specific study problem, which leads to natural selection of “*chromosomes as a combination of genes*” based on a defined “*function fitness,*” as a measure of the degree of adaptation of an organism to its environment; the bigger function fitness value means the better the organism fits and is adapted to the environment. In general, “*evolutionary algorithms*” focus only on a subset of mechanisms defined over the biological evolutionary process [18]. The main natural methods for artificial evolution are based on: “*reproduction,*” “*mutation,*” “*recombination,*” “*natural selection,*” and “*survival*”:

- “*Reproduction*” is a process of making a copy of a gene.
- “*Mutation*” is the ability to change the structure of a chromosome.
- “*Recombination*” is the genetic rearrange in the “*chromosome.*”
- “*Natural selection*” is a tool to choose based on better adaptation to their environment.
- “*Survival*” is a reward for continued living and produces more offspring.

The main objective of “*Evolutionary algorithms*” is to use natural mechanisms to try to find solutions for computationally complex problems for optimization for a single objective and/or multiobjective. “*Evolutionary Algorithms*” are more frequently used in many “*AI applications,*” especially for “*search and optimization.*” The more frequently algorithms used are “*Genetic Algorithms,*” “*Genetic Programming,*” “*Particle Swarm Optimization,*” “*Differential Evolution,*” “*Ant Colony Optimization,*” “*Memetic algorithms,*” and others:

- “*Genetic Algorithms (GA)*” are used for finding optimized solutions to search problems based on “*Darwinian evolution for the theory of natural selection and evolutionary biology.*” “*GA*” are excellent for searching through large and complex datasets.
- “*Genetic programming (GP)*” is a model of programming based on “*biological evolution to handle a complex problem.*” Of a number of possible programs or functions of a larger application, the most effective programs survive and compete or crossbreed with other programs to continually approach closer to the needed solution. “*GP*” is also used for “*finding a mathematical formula for relationship among variables based on experimental data.*”
- “*Particle Swarm Optimization (PSO)*” algorithms are inspired by some mechanisms of various living species. “*PSO*” uses “*Swarm Intelligence*” to solve problems that can be represented as a point or surface in a multi-dimensional space. The swarm is given random starting solutions first instead of exploring the entire parameter

space, for this reason it is less sensitive to the problem of local minima. Candidate solutions known as “particles” are placed in the n-space and given an initial velocity and communication channel between the particles. A swarm of simple formulas moves these particles through the solution space and evaluates the quality of the solution according to some fitness criterion after each time step. Finally, the particles accelerate toward those particles within their communication group that have the highest fitness values. This is a similar behavior as the flocking of birds or the movement of schools of fish. Therefore “PSO” algorithms are referred to as nature-inspired or bioinspired algorithms.

- “Differential Evolution (DE)” adapts the “search during the evolutionary process based on vector differences.” “DE” optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality; it is known as “metaheuristics” as they make few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions but do not guarantee an optimal solution.
- “Ant Colony Optimization (ACO)” is based on the concept of “Artificial Ants” using multiagent methods inspired by the behavior of real ants. “ACO” is a probabilistic technique for solving computational problems to “find the shortest path on a weighted graph.” “ACO” is used for path optimization problems as vehicle routing and internet routing.
- “Memetic algorithms (MA)” are extensions of the traditional “Genetic Algorithm,” using a local search

technique to reduce the likelihood of premature convergence. Most “MA” algorithms can be interpreted as a search strategy in which a population of optimizing agents cooperate and compete using local strategies.

- Other non-Darwinian algorithms are *Baldwinian Evolutionary Algorithms*, *Lamarckian Evolutionary Algorithms*, *Genetic Local Search*, etc.

The main objective of “Evolutionary algorithms” is to use natural mechanisms to try to find solutions for computationally complex problems for optimization of a single objective and/or multiobjective. “Evolutionary Algorithms” are more frequently used in many “AI applications,” especially for “search and optimization.” The more frequently used algorithms used are “Genetic Algorithms,” “Genetic Programming,” “Particle Swarm Optimization,” “Differential Evolution,” “Ant Colony Optimization,” “Memetic algorithms,” and others.

3.3.1 A typical evolutionary algorithm

From “Biology” we know that “cells” are the basic building block of all living things. Therefore in each “cell,” there is the same set of “chromosomes,” and each one is basically strings of “DNA (deoxyribonucleic acid),” as shown in Fig. 3.1. “DNA” is a self-replicating material that is present in nearly in all living organisms as the main constituent of “chromosomes and it is the carrier of genetic information.”

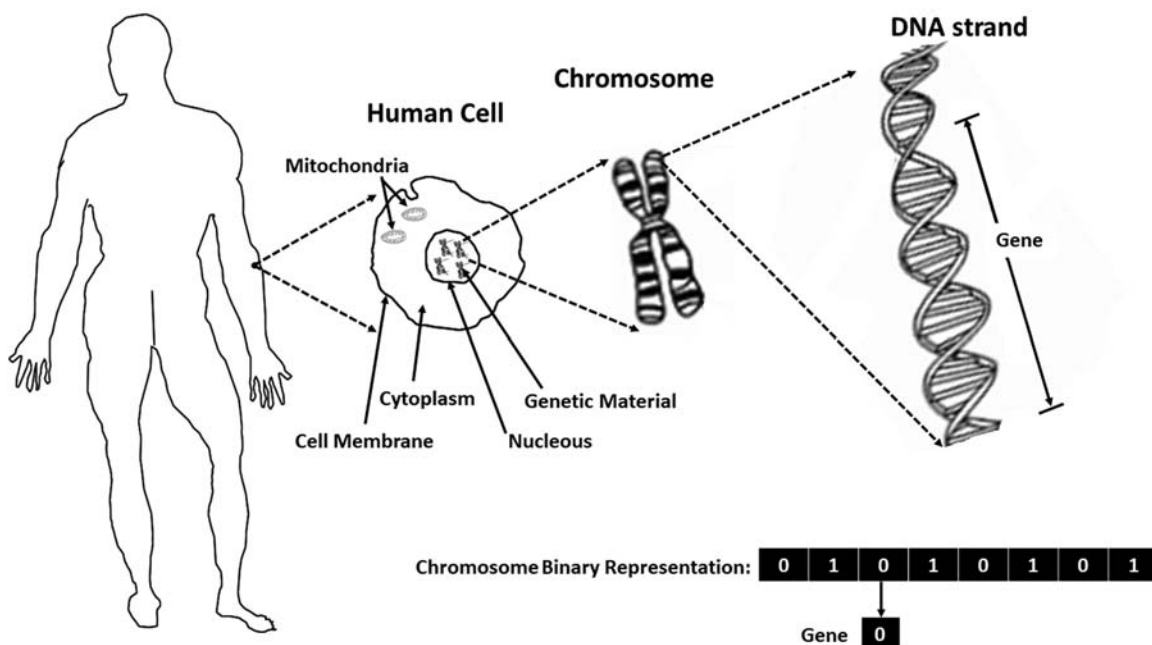


FIGURE 3.1 The human body is built with different types of cells that have the same set of chromosomes with strings of DNA which carry the genetic information.

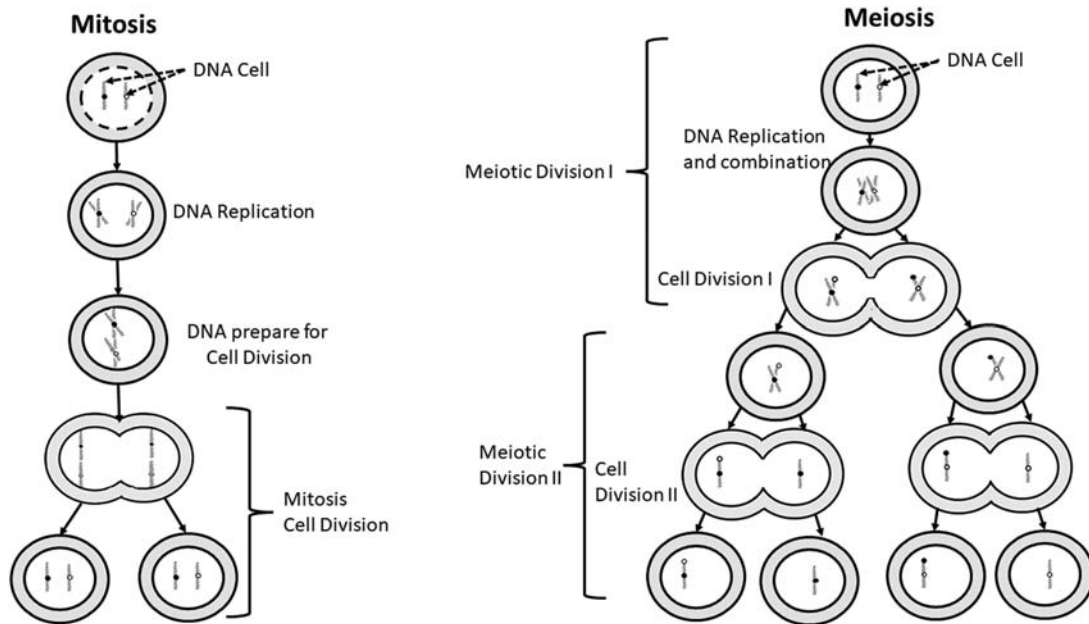


FIGURE 3.2 The reproduction of genetic information can be: “Mitosis,” which is the normal way of growing “multicell structures like organs,” and “Meiosis,” which is the basic sexual reproduction that creates “gametes, or sex cells as eggs or sperm.”

The “Genes” are encoded in the “chromosomes,” and every “gene” has a unique position on the “chromosome.” The entire combination of “genes” is called the “genotype” and each one leads to a “phenotype” that is a set of observable characteristics of an individual, such as eye color, height, “disease predisposition,” etc. The “phenotype” is affected by changes to the underlying genetic code, and this can be selected in the “reproduction as genetic information.” Basically, the reproduction of the genetic information can be of two types: “Mitosis” or “Meiosis” as indicated in Fig. 3.2:

1. “Mitosis” is copying the same genetic information to a new offspring and “there is no exchange of information.” “Mitosis” is the normal way of growth for “multicell structures like organs,” as shown on the left side of Fig. 3.2.
2. “Meiosis” is the basis of sexual reproduction. After the “meiotic” division two “gametes” appear. A “gamete” is a mature “haploid,” which is a single cell of unpaired chromosomes, and it can be a male or female germ cell that is able to unite with another of the opposite sex. In sexual reproduction two “gametes” conjugate to form a “zygote,” which will become the new individual. There are options in the reproduction process such as “Crossover” that leads to new “genotypes” and “Mutations” which are errors of single point mutations that are quite common in the copying process.

The steps of a typical “Evolutionary Algorithm” generate solutions to “AI Optimization in BME” problems using techniques inspired by natural evolution, such as “Genetic

Algorithms,” is shown in Fig. 3.3. The description of each step is:

- “Step 1) Method of representation” is the “encoding of parameters,” the most common are: binary strings, array of integers (usually bound), arrays of letters, etc.
- “Step 2) Create initial population” usually is assigned using a random generator.
- “Step 3) Method of selection” is usually: “Fitness Proportionate Selection,” “Roulette-wheel selection,” “Elitist selection,” “Cutoff selection,” and other fitness function. Where each description is:
 - “Fitness Proportionate Selection”: each individual can become a parent with a probability which is directly proportional to its fitness.
 - “Roulette-wheel selection”: this can be represented as a game of roulette, where everyone gets a slice of the wheel, but more fit ones get larger slices than less fit ones. A “Fitness function” assigns a fitness to possible solutions or “chromosomes” associated with the probability of selection. It is calculated as indicated in Eq. (3.2).

Fitness function for Roulette – wheel

$$FP = \frac{F_i}{\sum_{i=1}^n F_i} \quad (3.2)$$

- “Elitist selection” chooses only the most fit members of each generation
- “Cutoff selection” selects only those that are above a certain cutoff for the target function.
- Other fitness function can be deduced as needed.

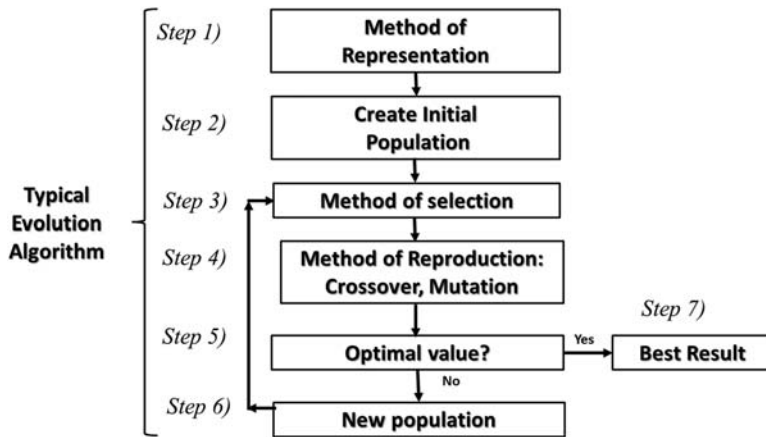


FIGURE 3.3 The typical steps of an “Evolutionary Algorithm.”

- “Step 4) Method of reproduction” known also as “Evolution.” There are two primary reproduction methods: “Crossover” and “Mutation”:
 - “Crossover” is when the chromosomes of the two parents are copied randomly to the next generation to form new offspring. Other ways are: “uniform crossover” that occurs when a random subset is chosen, and then the subset is taken from parent “1” and the other bits from parent “2”; or “multipoint crossover” where a specific range is copied from parent “1” and substituted onto the copy of parent “2.”
 - “Mutation” is an error that can occur in the copying process, the most common is a single point mutation, but other kinds of errors can also occur, such as deletion, inversions, substitution, etc.
- “Step 5) Optimal value?” If yes, then we have then the best answer and skip to step 7), if not continue the algorithm.
- “Step 6) New Population” is the result of method of reproduction applied.
- “Step 7) Best result” is the final answer, which can be the optimum found or when the time limit is reached.

The seven steps for a typical “Evolutionary Algorithm” to generate solutions to “AI Optimization in BME” problems using techniques inspired by natural evolution, such as the “Genetic Algorithms,” are shown in the Fig. 3.3.

3.3.2 Genetic algorithms for AI optimization in BME

“Genetic Algorithms (GA)” are frequently used for “AI Optimization in many fields of BME for many AI applications,” as explained in Section 3.2. The next tutorial is a general example to explain the bases of a “GA” to visualize the objectives, procedure, and benefits of following this method.

3.3.2.1 Research 3.1 Genetic algorithm basic seven steps for selection by priorities

3.3.2.1.1 Problem

What items in a bag from a list are recommended to be taken to the hospital when a patient and his companion will stay for a month*? Note: All hospitals provide beds with sheets and blanket only for the patient but not for a companion.

Assume that a patient must spend a month in a hospital, by regulation all patients can carry a bag with a maximum weight of “30 kg.”, A specific patient and his companion are thinking to take their necessary items as indicated in Fig. 3.4*.

Note*: In “Computer Science” this kind of problem is known as a “knapsack-problem.”

3.3.2.1.2 Objective

Could we suggest using a “Genetic Algorithm (GA)” to determine which items are best to take in a bag to the hospital for a stay of a month a patient and his companion, where the selection of items is from a list based on the weight of each item and their assigned priority value for each item, as indicated in Fig. 3.4A.

3.3.2.1.3 Procedure

Applying the typical steps for an “Evolutionary Algorithm” as indicated in Fig. 3.3:

“Step 1) Method of representation.”

We decide to use “Binary representation of 6 positions,” where “1” indicates that the item is included in the bag and “0” indicates the item is not included. As stated in Fig. 3.4B.

“Step 2) Create initial population.”

It will consist of the creation of 4 initial “chromosomes,” as indicated in Fig. 3.4C, where each suggested “chromosome” has a total weight ≤ 30 kg.

“Step 3) Method of selection”

The “fitness function” to use is based on a “Roulette-wheel” obtained from the calculation of priority points

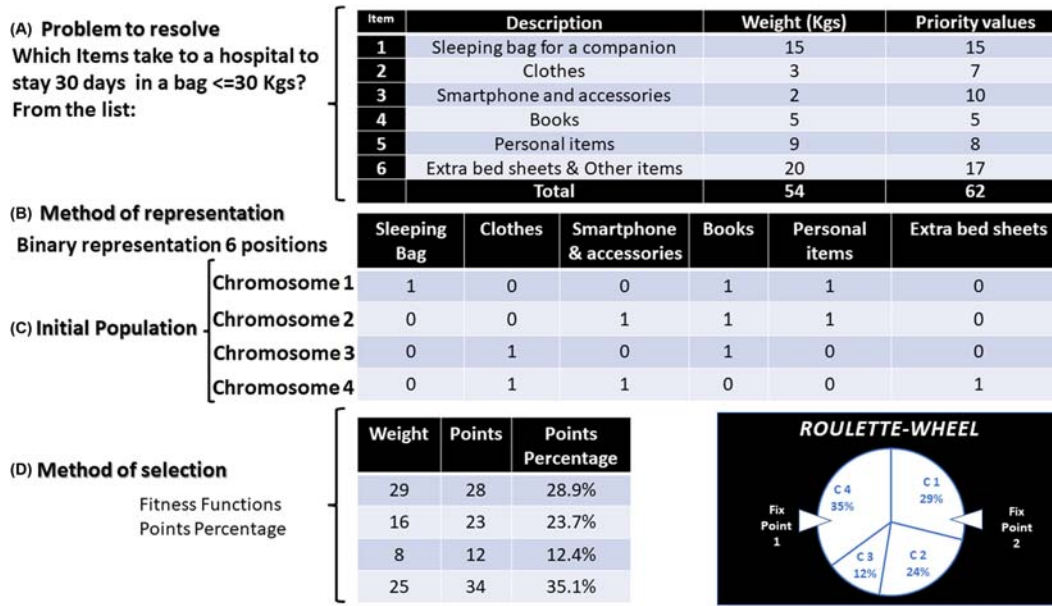


FIGURE 3.4 Example of a typical Evolutionary Algorithm. (A) Problem to solve, (B) method of representation, (C) create initial population, and (D) method of selection of “Roulette-wheel.”

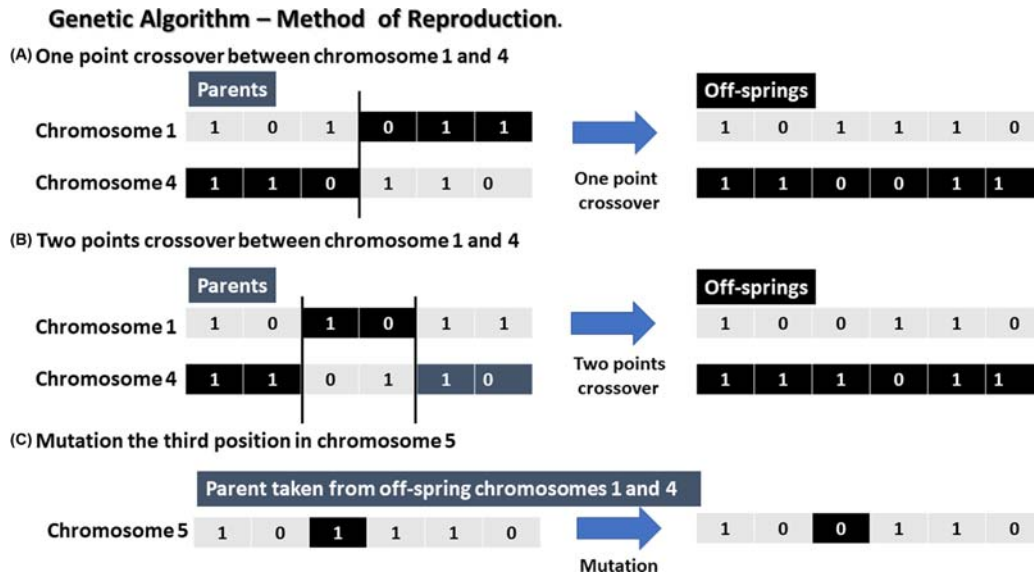


FIGURE 3.5 Example of Method of reproduction: (A) “One-point crossover” between chromosome 1 and 4, (B) “Two points crossover” between chromosome 1 and 4, and (C) “Mutation” of the third position of parent taken from offspring chromosomes 1 and 4.

for each item in the “chromosome” divided by the total priority of “all chromosomes” as indicated in Eq. (3.2) and the results for each “chromosome” in Fig. 3.4D.

“Step 4) Method of reproduction”

The most common methods for “reproduction in GA” are:

- “One-point crossover” between chromosome 1 and 4, as shown in Fig. 3.5A.
- “Two points crossovers” between chromosome 1 and 4, as shown in Fig. 3.5B.

- “Mutation” of the third position form a parent taken from offspring chromosomes 1 and 4, as shown in Fig. 3.5C.

“Step 5) Optimal value?”

Basically, the “optimal value” is reached when one of the different termination conditions exist, these are:

- There is no improvement in the population for over x iterations.
- The predefined absolute number of generations for our algorithm is reached.

- The “*Fitness function*” has reached a predefined value.

If the optimal value is reached then go to step 7), otherwise continue with the following step.

“*Step 6) New Population*”

“*New population*” is the result of method of “*reproduction in GA*” applied in the iteration, it is the result of the “*mutation*.” Then, go to step 3) over and over until reaching the “*optimal value*” described in step 5).

“*Step 7) Best result obtained*”

In “*GA*” an answer is always obtained, and it could be improved with time.

Conclusions

The Benefits of “*Genetic Algorithms*” are:

- The concept is easy to understand and it is very flexible, using building blocks that can be used in hybrid applications with “*Machine Learning*.”
- It is easy to exploit previous or alternate solutions.
- It supports multiobjective optimization.

3.3.3 Genetic algorithm for AI optimization in BME under MATLAB[®]

3.3.3.1 Research 3.2 Implementing genetic algorithm for AI optimization in BME with MATLAB

3.3.3.1.1 Problem

The tutorial implements in MATLAB the problem explained in Research 3.1: “*Which items to take to a hospital for a 30 day stay in a bag of ≤ 30 kg from a specific list*?*.” Note*: All hospitals provide beds with sheets and blanket only for the patient but not for a companion.

3.3.3.1.2 General objective

Obtain a list of which items are best to take to the hospital in the bag based on the weight of each item and their assigned priority value for each item, as indicated in Fig. 3.4A. Implement in MATLAB by applying the “*Global Optimization toolbox*” that includes a function solver for “*Genetic Algorithm*.” The general objective of the MATLAB “*ga()*” function solver is to find a minimum or maximum of function using the “*Genetic Algorithm*.”

3.3.3.1.3 Specific objectives

- Use the function “*ga()* from *Global Optimization Toolbox*” to find the minimum or maximum values.
- Specify “*Fitness function*,” “*Constraints function*” and understand options available for “*GA*.”
- Create their respective user MATLAB functions.

- Run and interpret the results to find the recommended items to take to the hospital under optimization of values and priorities.

3.3.3.1.4 Background

Please review Research 3.1 Genetic algorithm basic seven steps explained in figure 3.3.

3.3.3.1.5 Dataset

The items that we can take to stay at the hospital are shown in Fig. 3.4A.

3.3.3.1.6 Procedure

A function solver for “*Genetic Algorithm*” named “*ga()*” allows one to resolve problems for mixed-integer or continuous variable optimization, constrained or unconstrained, to find the minimum or maximum of a function. The most important parameters to specify in the “*GA*” algorithm are “*Fitness function*,” “*Constraints Function*,” and “*understand options available*”:

- “*Fitness function*” defines the more suitable value to fulfill a particular role or task. For this tutorial research, the following condition must be evaluated as indicated in Eq. (3.3).

Fitness function for bag to take to hospital fitness

$$= \sum_{i=1}^n c_i v_i \quad (3.3)$$

where n = chromosome length, c_i = i th gene, and v_i = i th weight.

Note: For the example: $n = 6$, c = chromosome of 6 bits, and weight is a vector with the values for each possible item that can be in the bag to take to the hospital, and it is indicated as weight = [15 3 2 5 9 20] taken from Fig. 3.4A.

The MATLAB implementation for the “*Fitness function*” implementing Eq. (3.3) is shown in Table 3.1.

- “*Constraints function*” defines the limitation or restriction of the *Fitness function* as indicated in Eq. (3.4).

Constraints function for bag hospital constraints

$$= \sum_{i=1}^n c_i w_i \leq bw \quad (3.4)$$

where n = chromosome length, c_i = i th gene, w_i = i th weight, and bw = bag maximum weight.

For the example: “ $n = 6$,” “ c = chromosome of 6 bits,” and value is a vector with the priority values for each possible item in the bag to take to the hospital, such as: “ $value = [15 7, 10 5 8 17]$,” as shown in Fig. 3.7B.

TABLE 3.1 The MATLAB implementation for the “Fitness function” as indicated in Eq. (3.3) ^a.

```
% Fitness function for Genetic Algorithm for Chapter 3, Example 3.3.3.1
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
function y = fitness(x)
global value; % use de array of integer for the priority value of each item
y = -value*x'; % Find the maximum of the fitness using the negation symbol
```

^aThis MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_hospital\fitness.m.”

TABLE 3.2 The MATLAB implementation for the “Constraints function as indicated in Eq. (3.4) ^a.

```
% Constraints function for Genetic Algorithm for Chapter 3, Example 3.3.3.1
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
function [c, ceq] = constraints(x)
global weight;
c = -weight*x'; % Find the maximum of the Constraints using the negation symbol
ceq = [];
```

^aThis MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_hospital\constraints.m.”

The MATLAB implementation for the “Constraints function” implementing Eq. (3.4) is shown in Table 3.2.

- *Understand options available.* The two more important main options available in the function solver “ga()” are “*optimoptions*” that create optimization options and “*resetoptions*” that reset options to their default values.

To resolve the problem we can find the minimum or maximum of the function using the “Genetic Algorithm ga()” in the “MATLAB Global Optimization Toolbox”; specifying the variables following the general syntax and the definition for its input and output parameters, as indicated in Fig. 3.6.

The implementation for the MATLAB main program is shown in Table 3.3, where the “ga()” function solver parameters are indicated in Fig. 3.6C.

At the end of the main program listed in Table 3.3 a function “*itemsToTake(x)*” is called to create the list of items that are better to take to the hospital based on the weight of each item and their assigned priority value for each item. This function “*itemsToTake(x)*” is listed in Table 3.4.

3.3.3.1.7 Results

When the main program shown at Table 3.3 is run, a chart is generated to indicate the best and the mean values of the “function Constraints,” as shown in Fig. 3.7A. This

chart indicates that the best solution is stable after the six generation of the 10th request. Finally, in the MATLAB command windows the answer is listed and indicates that the final best chromosome recommended is equal to [1 1 1 0 1 0]. This “chromosome” has a “weight = 29” (must be ≤ 30) and “priority value = 40.” The items that we recommend to take to the hospital are listed in Fig. 3.7B, these are: “Sleeping bag, Clothes, Smartphone, and Personal items.” All the MATLAB results for this tutorial are shown in Fig. 3.7C.

Conclusion

Applying the “Genetic Algorithm” through the function solver “ga()” available in “MATLAB Global Optimization Toolbox” allows one to develop a solution for this algorithm in an efficient way. Besides, this “Genetic Algorithm” function solves smooth or nonsmoothed optimization problems with any type of constraint that is based on a stochastic process that searches randomly by mutation and crossover among population members.

3.3.4 General analysis and optimization of 2D and 3D data in biomedical engineering

Generally, numeric data in biomedical engineering usually is frequently obtained with two variables (2D), three variables

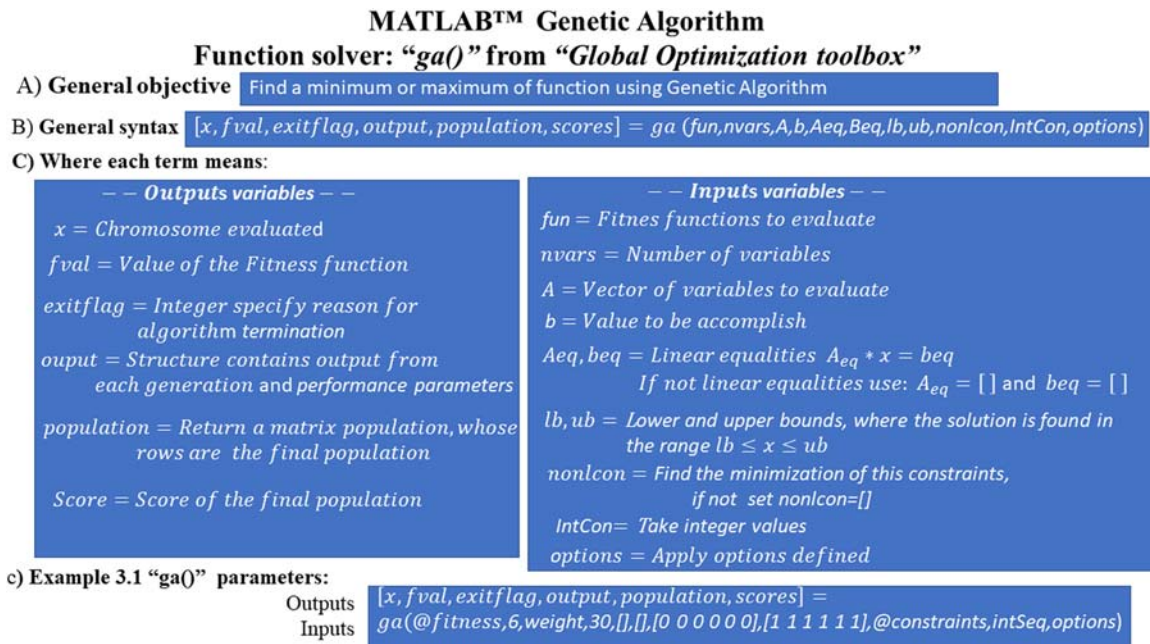


FIGURE 3.6 Genetic Algorithm implemented by function solver “ga()” in MATLAB Global Optimization toolbox.

(3D), or more variables. These datasets represent the measurement values that can be obtained from different kinds of Bioinstruments, such as “*Electromyography (EMG)*,” “*Ground Reaction Forces (GRF)*,” “*Electrocardiograph (ECG)*,” “*Electroencephalography (EEG)*,” etc. These values come from measurements by sensors in the human body, such as electric current, voltage, heat, pressure, etc. against time, range of frequency, space etc., these values can be handled as a vector or data matrix. Also, the imaging datasets can come from imaging bioinstruments systems, such as “*CT Scanners*,” “*Ultrasound machines*,” “*Magnetic resonance imaging (MRI)*,” “*X-rays*,” etc., which deliver a set of images showing the human body’s internal parts on different scales, that is, microscopic, macroscopic, etc. These images can be: “*2D cut slices*,” “*3D volumetric images*,” “*4D additional time dimension*,” or “*4D–5D with images from multiple channels*”*.

Note*: This chapter has as an objective to apply Artificial Intelligence models based on 2D and 3D using measurement values as vectors and/or a data matrix; others dimensions and types are covered in the following chapters: Chapter 4, *Machine learning models for numeric data analysis*; Chapter 5, *Deep learning models for images analysis*; and Chapter 6, *Cognitive Computing models for speech, and text analysis*.”

A typical algorithm for Optimization and AI numeric data numeric analysis applying “Evolutionary Algorithms”

in biomedical engineering can be summarized as indicated in Fig. 3.8. The steps are as follows:

“*Step 1) Data Inspection*”

The data inspection allows one to verify that the data is valid by applying statistics or previsualizations to detect anomalies, for example, “*detect outliers*” are data points that differ significantly from the other observations (usually obtained as experimental errors), “*apply normalization*” normalizes data by eliminating the units of measurement, enabling easy comparison of the data with other experiments, etc.

“*Step 2) Obtain Mathematical Model by Fitting data*”

Data fitting is the process of fitting models to data using: polynomials, curves, surfaces, and nonparametric methods. Data fitting allows a general data analysis to achieve optimization, and prediction.

“*Step 3) Evaluate model with Residuals*”

This is a must, always check the residuals of the mathematical model obtained by fitting data with the real data, the least residual value represents better precision in the data analysis.

“*Step 4) Define a Fitness Function*”

A “*Fitness function*” is an objective function that assigns a fitness to possible solutions associated with the probability of the selection.

“*Step 5) Apply Evolutionary Algorithm*”

There are many “Evolutionary Algorithms” that allow obtaining solutions for optimization, besides the traditional search functions.

“*Step 6) Compare results and select optimal result*”

TABLE 3.3 Main program for the solution using a Genetic Algorithm implemented by function solver “ga()” in MATLAB Global Optimization Toolbox.

```

% Genetic Algorithm
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
% A patient has to spend a month in a hospital, by regulation all patients
% can carry a bag with a maximum weight of 30 Kg. A patient is thinking
% to take his necessary items as indicated in figure 3.7a.
% Could we suggest using a “Genetic Algorithm” which item are best to take
% to the hospital in the bag based on the weight of each item and their
% assigned priority points for each item?
% IMPORTANT: This problem use the MATLAB "Global Optimization toolbox"

%% Define variables needed
rng(1,'twister'); % Random generator using Mersenne Twister with seed 1.
intSeq= 1:6; % Generate integer sequence from 1 to 6, to identify each item
global weight; % Global array of integers for weight of each item
global value; % Global array of integers for values of priority for item
global items; % Global array of string for items
weight=[15 3 2 5 9 20]; % Weight of each item
value=[15 7,10 5 8 17]; % Values for priority of each item
items=[" Sleeping bag "," Clothes "," Smartphone "," Books "," Personal items "," Extra bed sheets "];

%% Define ga() options
options = optimoptions('ga'); % Default options
% 10 Generations of 100 chromosomes
options = optimoptions(options,'MaxGenerations', 10);
options = optimoptions(options,'MaxStallGenerations', inf);
options = optimoptions(options,'FunctionTolerance', 0);
options = optimoptions(options,'ConstraintTolerance', 0);
options = optimoptions(options,'Display', 'off');
options = optimoptions(options,'PlotFcn', { @gaplotbestf });

%% Run the Genetic Algorithm function
[x,fval,exitflag,output,population,score] = ...
ga(@fitness,6,weight,30,[],[],[0 0 0 0 0 0],[1 1 1 1 1 1],@constraints,intSeq,options);
disp(['Final chromosme = ',num2str(x)]);
disp(['Weight each item= ',num2str(weight)', Total weight= ',num2str(weight*x)']);
disp(['Value each item= ',num2str(value)', Total value = ',num2str(value*x)']);
disp('Recommended items to take to the hospital= ');
itemsToTake(x);

```

Note: This MATLAB program can be downloaded from the website companion and installed in the following path “. \Exercises_book_ABME\CH3\GA_Matlab\GA_hospital\ga_hospital.m.”

TABLE 3.4 Function “itemsToTake(x)” is used to create the list of items that are better to take to the hospital using a Genetic Algorithm.

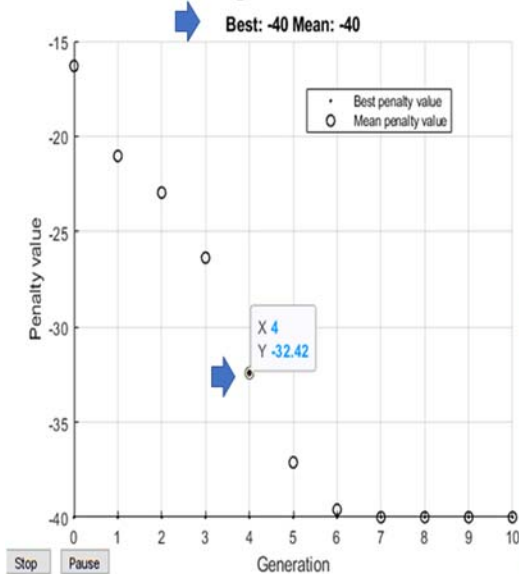
```

% itemsToTake function for Genetic Algorithm for Chapter 3, Example 3.3.3.1
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
function itemsToTake(x)
global items;
final = length(x);
for i = 1:final
    if x(i) == 1
        disp(items(i));
    end
end
end

```

Results for example 3.1 What items in a bag are recommend to the hospital? using MATLAB Genetic Algorithm

(A) Chart that show the best and mean penalty (B) Problem to resolve: Which Items take to a hospital? Value after the 6th generation .



Item	Description	Weight (Kgs)	Priority values
1	Sleeping bag for a companion	15	15
2	Clothes	3	7
3	Smartphone and accessories	2	10
4	Books	5	5
5	Personal items	9	8
6	Extra bed sheets & Others items	20	17
Total		54	62

(C) Using Genetic Algorithm recommend to take the following items:

```

Command Window
>> ga_hospital
Final chromosome = 1 1 1 0 1 0
Weight each item= 15 3 2 5 9 20, Total weight= 29
Value each item= 15 7 10 5 8 17, Total value = 40
Recommended items to take to the hospital=
Sleeping bag
Clothes
Smartphone
Personal items
fx >> |
    
```

FIGURE 3.7 Results for example 3.1 applying MATLAB Genetic Algorithm: (A) penalty chart, (B) original list to take in a bag ≤ 30 kg, and (C) final chromosome with total weight, total priority values and list if item to take.

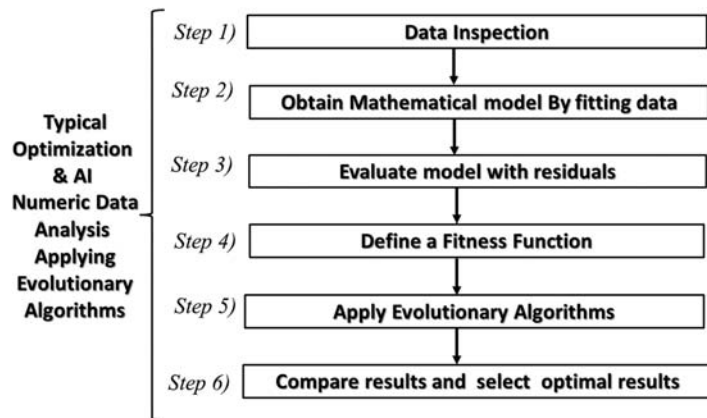


FIGURE 3.8 A typical algorithm for Optimization and AI numeric data analysis applying “Evolutionary Algorithms” in biomedical engineering.

It is always recommended to use more than one method to assure the optimization result is global and not a local result.

Tutorials for data optimization using MATLAB are explained in the two next sections:
 Section 3.5 2D numeric data using MATLAB Global Optimization Toolbox.
 Section 3.6 3D numeric data using MATLAB Global Optimization Toolbox.

3.3.5 MATLAB analysis and optimization of “2D” data in biomedical engineering

3.3.5.1 Research 3.3 Analysis and optimization of “2D” data of “Body measurement of nerve contractions” applying MATLAB

3.3.5.1.1 Problem to resolve

“Body measurement of nerve contractions” from a specialized “biomedical instrument” were obtained as vector values in “mVolts” versus a “time vector” in “sec.” As part of “BME research” there is a need to find the “global

minimum value” that represents the lower contraction of the nerve during the test.

3.3.5.1.2 General objective

Use general analysis and the six steps of “*optimization of 2D using Evolutionary Algorithms*,” as shown in Fig. 3.8, to obtain a “*Mathematical Polynomial Model*” by fitting the dataset values, and then apply the “*MATLAB Global Optimization Toolbox*” to find the solution to the problem requesting the “*global minimum value*,” which represents the “*lower contraction of the nerve during the test*.”

3.3.5.1.3 Specific objectives

- Use general analysis and “*optimization of 2D using two Evolutionary Algorithms*”:
 - “*Genetic algorithm*”
 - “*Particle Swarm Algorithm*”
 - and two “*traditional search algorithms*”:
 - “*Direct search Algorithm*” and
 - “*Global search Algorithm*.”
 - Obtain a “*Mathematical Polynomial Model*” by fitting the data and then applying the “*MATLAB Global Optimization Toolbox*” to find the solution to the problem requested.
 - Compare the results of the four methods used to resolve the problem.

3.3.5.1.4 Dataset

These are the vector values: for “*time t = [0 0.25 0.50 0.75 1 1.25 1.50 1.75 2 2.25]*,” and their respective vector values for “*Body measurement of nerve contractions y = [0.85 0.675 0.85 1.01 1.15 1.35 1.45 1.25 1.35 1.45]*.”

3.3.5.1.5 Procedure

Apply MATLAB following the steps for a “*Typical algorithm for optimization & AI numeric data*,” as indicated in Fig. 3.8, to obtain the solution requested. The MATLAB solution is based on: “*main program*,” “*four subprograms*,” and “*one user function*,” where each description is explained as:

- “*Main program*” implements steps 1–3 of the algorithm shown under the name “*Optim_Analysis_2D.m*” with a user MATLAB program. It is listed in Table 3.5, where:
 - “*Step 1) Data Inspection of Original discrete data*” generates a “*2D chart showing the original discrete vectors values*,” as indicated on the left of Fig. 3.9.
 - “*Step 2) Mathematical Model by fitting data*” that generates a “*fifth-degree polynomial model*,” and generates a “*2D Chart showing the discrete*

original vector values and the continuous polyline model” as indicated in the center Fig. 3.9.

- “*Step 3) Evaluate model with residuals*” that generates a “*chart showing the residuals values*” as indicated on the right of Fig. 3.9.
- A user function: “*polyline_fitness.m*” listed in Table 3.6 that processes:
 - “*Step 4) Define Fitness Function*,” which is needed in the “*Evolutionary Algorithms*,” and evaluate the polyline shown in Eq. (3.5).

Polyline for Fitness Functions Analysis and optimization of “2D” data in BME

$$\begin{aligned} \text{fitnes function} = & 0.0775x^5 + 0.0689x^4 \\ & - 1.5932x^3 + 3.0054x^2 - 1.1943x + 0.8428 \end{aligned} \quad (3.5)$$

- “*Genetic Algorithm*” is called in the subprogram “*ga_polyline.m*” as listed in Table 3.7. It processes:
 - “*Step 5) Apply Evolutionary Algorithm: Genetic algorithm*” defining the parameters needed to call the function “*ga()*” available on “*MATLAB Global Optimization Toolbox*.” It generates the left-hand chart of Fig. 3.10, and
 - “*Step 6) Compare results and select optimal results*”: the results are shown in Fig. 3.10A.
- “*Particle Swarm Algorithm*” is called by the subprogram “*ps_polyline.m*” listed in Table 3.8. It processes:
 - “*Step 5) Apply Evolutionary Algorithm: Particle Swarm Algorithm*” defining the parameters needed to call the function “*particleswarm()*” available on *MATLAB Global Optimization Toolbox*,” and
 - “*Step 6) Compare results and select optimal results*”: it calculates the results as shown in Fig. 3.10B.
- “*Direct search Algorithm*” is called by the subprogram “*ds_polyline.m*” listed in Table 3.9. It processes:
 - “*Step 5c) Apply Algorithm: Direct search*” defining the parameters needed to call the function “*patternsearch ()*” available on *MATLAB Global Optimization Toolbox*.” It generates the chart shown in right side of Fig. 3.10, and
 - “*Step 6) Compare results and select optimal results*”: it calculates the results shown in Fig. 3.10C.
- “*Global search Algorithm*” is called by the subprogram “*gs_polyline.m*” listed in Table 3.10. The process is:
 - “*Step 5) Apply Algorithm: Global search*” defining the parameters needed to call the function “*globalsearch ()*” available on *MATLAB Global Optimization Toolbox*,” and
 - “*Step 6) Compare results and select optimal results*”: it calculates the results shown in Fig. 3.10D.

TABLE 3.5 Main program: “*Optim_Analysis_2D.m*” for MATLAB Analysis and optimization 2D data^a.

```

% Main Program “Optim_Analysis_2D.m”
% 3.2.4 MATLAB™ Analysis & optimization 2D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa

%% Step 1) Data Inspection of Original discrete data
t=[0 .25 .50 .75 1 1.25 1.50 1.75 2 2.25];
y=[0.85 0.675 .85 1.01 1.15 1.35 1.45 1.25 1.35 1.45];
plot(t,y,'x');title(" Step 1) Plot of Data inspection (Points)")
xlabel("Time (Seconds)");ylabel("Biomedical Signal ( mVolts)");

%% Step 2) Mathematical Model by fitting data
% Modeling this data using a fifth-degree polynomial function,
p = polyfit(t,y,5)
% Plot original data and model on the same plot.
t2 = 0:0.1:2.8;
y2 = polyval(p,t2);
figure
plot(t,y,'x',t2,y2)
title('Step 2) Plot of Data inspection (Points) and Model (Line)');
xlabel("Time (Seconds)");ylabel("Biomedical Signal (mVolts)");
legend("Original points","Polyline Model");
disp(strcat('Polyline Coefficients = ',num2str(p)));

%% Step 3) Evaluate model with residuals
% Evaluated at data time vector
y2 = polyval(p,t);
% Calculate the residuals.
res = y - y2;
% Plot the Mathematical Model residuals.
figure, plot(t,res,'+')
title('Step 3) Evaluate model using Residuals')
xlabel("Time (Seconds)");ylabel("Biomedical Signal (mVolts)");
disp(strcat('Model Residuals from + ',num2str(max(res)), ' to ', num2str(min(res))));

```

^aThis MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_polyfit\GA_polyfit\Optim_Analysis_2D.m.”

Conclusion

Four methods were used to calculate the optimization requested to “find the minimum value.” The results are summarized at the bottom of Fig. 3.10, indicated as: “Step 6) Compare results and select optimal results.” Where the two of them using “Evolutionary Algorithms: Genetic Algorithm and Particle Swarm Algorithm” show very similar results of: “Minimum found at time = 0.245 with a Fitness value = 0.707,” and two more using “Traditional algorithms for search: Direct search and Global search” with similar optimization results, then we can be sure that “the result obtained is the global minimum.”

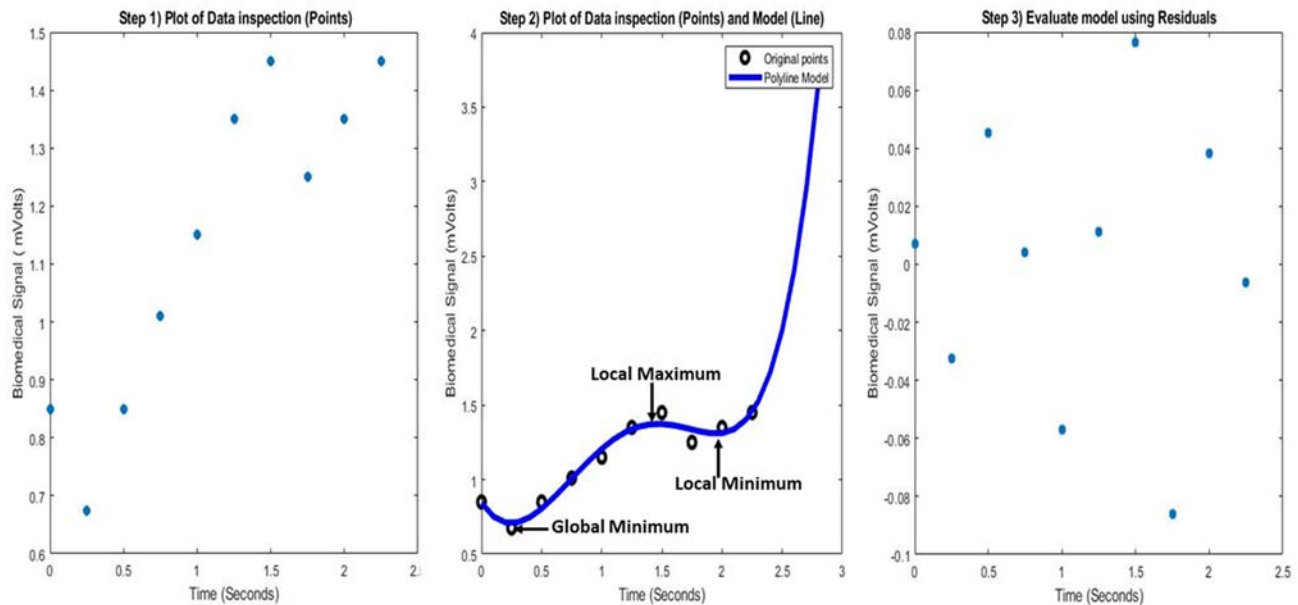
3.3.6 MATLAB analysis and optimization of 3D data in biomedical engineering

3.3.6.1 Research 3.4 Analysis and optimization of “3D” data for “the center of mass of an upper extremity—right arm movement from a patient” applying MATLAB

3.3.6.1.1 Problem to resolve

A serial of measurements of three dimension values that represent “the center of mass of an upper extremity—right arm movement from a patient,” were obtained from a “biomedical instrument” as three vectors, each one with a size of [295 1], representing each coordinate as “x,” “y,” and “z.” The dataset must be analyzed to detect the

3.2.4 MATLAB™ Tutorial Analysis and optimization of “2D” data in Biomedical Engineering



Step 4) Define a Fitness Function

Polyline for Fitness Functions =
 $0.0775x^5 + 0.0689x^4 - 1.5932x^3 + 3.0054x^2 - 1.1943x + 0.8428$

Model Residuals from +0.076266 to -0.086217

FIGURE 3.9 MATLAB example for analysis and optimization of “2D” data in biomedical engineering: step 1) Plot of Data inspection, step 2) Plot of Model obtained by fitting data and original data values, and step 3) Plot showing the residuals values of the model compared to the original values.

TABLE 3.6 Function: “polyline_fitness.m” for MATLAB Analysis and optimization 2D numeric data.

```
% Function polyline_fitness.m
% 3.2.4 MATLAB Analysis & optimization 2D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
function y = polyline_fitness(x)
%Step 4) Define Fitness Function
y = 0.0775*x(1)^5 + 0.0689*x(1)^4 - 1.5932*x(1)^3 + 3.0054*x(1)^2 - 1.1943*x(1) + 0.8428;
```

Note*: This MATLAB program can be downloaded from the website companion and installed in the following path “. . . \Exercises_book_ABME\CH3 \GA_Matlab \GA_polyfit \GA_polyfit\polyline_fitness.m.”

“global minimum” that represents the “optimal value” to be used for other “data analysis using AI algorithms.”

3.3.6.1.2 General objective

Use six steps of “general analysis and optimization of 3D” applying “Evolutionary Algorithms” with “MATLAB Global Optimization Toolbox” to find the solution to the problem. Here we use two “Evolutionary Algorithms” and two “traditional search algorithms”: “Direct search Algorithm” and “Global search Algorithm” “to find the solution to the problem requested the “global minimum value,” that

represents the 3D coordinates for the “center of mass of an upper extremity—right arm movement from a patient.

3.3.6.1.3 Specific objectives

- Use “general analysis” and “optimization of 3D using two Evolutionary Algorithms”:
 - “Genetic algorithm,” and
 - “Particle Swarm Algorithm.”
- and two “traditional search algorithms”:
 - “Direct search Algorithm” and
 - “Global search Algorithm.”

TABLE 3.7 Subprogram to use “Genetic Algorithm”: “ga_polyline.m” for MATLAB Analysis and optimization 2D numeric data^a.

```
% Sub-program “ga_polyline.m”
% 3.2.4 MATLAB™ Analysis & optimization 2D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
disp('Step 5a) Apply Evolutionary Algorithm: Genetic algorithm ');
rng default % For reproducibility
FitnessFunction = @polyline_fitness;
numberOfVariables = 1;
lb = [0,0];warning('off','globaloptim:checkbound:lengthOfUpperBound');
ub = [.5,1.5];warning('off','globaloptim:checkbound:lengthOfLowerBound')
options = optimoptions('ga'); % Default options
% 10 Generations
options = optimoptions(options,'MaxGenerations', 10);
options = optimoptions(options,'MaxStallGenerations', inf);
%options = optimoptions(options,'FunctionTolerance', 0);
options = optimoptions(options,'ConstraintTolerance', 0);
options = optimoptions(options,'Display', 'off');
options = optimoptions(options,'PlotFcn', { @gaplotbestf });
[x,fval] = ga(FitnessFunction,numberOfVariables,[],[],[],[],lb,ub,[],options);
disp(strcat('ga() minimum found at: t in sec. = ',num2str(x)));
disp(strcat('          signal value = ', num2str(fval)));
```

^aThis MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_polyfit\ga_polyline.m.”

Continue... 3.2.4 MATLAB™ Tutorial Analysis and optimization of “2D” data in Biomedical Engineering

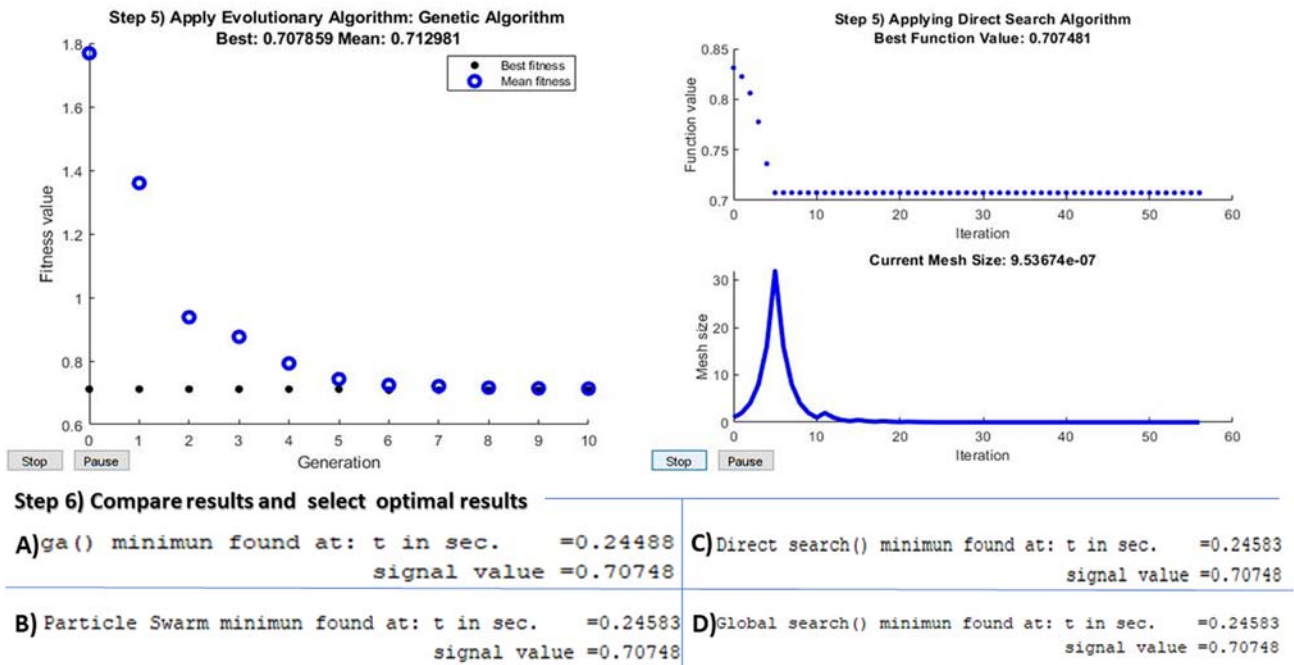


FIGURE 3.10 MATLAB example for analysis and optimization of “2D” data in Biomedical Engineering: step 5): upper left chart applying Genetic Algorithm, upper right charts applying Direct Search Algorithm, lower Plot of Data inspection, and lower table comparison between the four algorithms used for 2D optimization.

TABLE 3.8 Subprogram to use “*Particle Swarm Algorithm*”: “ps_polyline.m” from Section 3.3.5 MATLAB Analysis and optimization 2D numeric data^a.

```
% The subprogram “ps_polyline.m”
% 3.2.4 MATLAB Analysis & optimization 2D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
disp(“Step 5b) Apply Evolutionary Algorithm: Particle Swarm Algorithm”);
disp(“Particle swarm Polyline-----”);
rng default % for reproducibility
fun = @polyline_fitness; % objective
nvars = 1;
lb = [0,0];warning(“off”,“globaloptim:checkbound:lengthOfUpperBound”);
ub = [.5,1.5];warning(“off”,“globaloptim:checkbound:lengthOfLowerBound”);
options = optimoptions(“particleswarm”,“SwarmSize”,1000);
[x,fval,exitflag] = particleswarm(fun,nvars,lb,ub,options);
disp(strcat(“Particle Swarm minimum found at: t in sec. =”,num2str(x)));
disp(strcat(“signal value =”, num2str(fval)));
```

^aThis MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_polyfit\ps_polyline.m.”

TABLE 3.9 Subprogram to use “*Direct search Algorithm*”: “ds_polyline.m” from Section 3.3.5 MATLAB Analysis and optimization 2D numeric data.

```
% Sub-program “ds_polyline.m”
% 3.2.4 MATLAB™ Analysis & optimization 2D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa

disp(“Step 5c) Apply Algorithm: Direct search ");
options = optimoptions('patternsearch','PlotFcn',{@psplotbestf,@psplotmeshsize});
[x,fval] = patternsearch(@polyline_fitness,[.01],[],[],[],[],[],[],options);
disp(strcat('Direct search() minimum found at: t in sec. = ',num2str(x)));
disp(strcat(' signal value =', num2str(fval)));
```

Note*: This MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_polyfit\ds_polyline.m.”

- Obtain a “*Mathematical Polynomial Model*” by fitting the data and then applying the “*MATLAB Global Optimization Toolbox*” to find the solution to the problem requested.
- Compare the results of the four methods used to resolve the 3D optimization problem.

3.3.6.1.4 Procedure

A general analysis and optimization of 3D applying “*Evolutionary Algorithms*” with “*MATLAB Global Optimization Toolbox*” is going to be used to find the solution to the given problem following the steps indicated in Fig. 3.8 that describe a “*typical algorithm for optimization & AI numeric data.*”

The MATLAB solution for this problem is based on: “*a main program,*” “*a function fitness*” to be used for

“*Evolutionary Algorithms,*” and “*four subprograms,*” one for each algorithm to calculate the minimum value. Each one is described as follows:

- “*Main program*” was created with the name “*Optim_Analysis_3D.m.*” This is listed in Table 3.11 and this program’s process steps 1) to 3) are as follows:
 - “*Step 1) Data Inspection of Original discrete data*”: to observe the value vectors a 3D chart is generated that shows the original discrete vectors values as indicated on the left of Fig. 3.11. For these kinds of values with a wide range in the vectors it is recommended to “*normalized*” the values of the three axes before applying the next step.
 - “*Step 2) Mathematical Model by fitting data*”: here a “*Surface Polynomial function*” is calculated,

TABLE 3.10 Subprogram to use “Global search Algorithm”: “gs_polyline.m” from Section 3.3.5 MATLAB Analysis and optimization 2D numeric data.

```

% Sub-program “gs_polyline.m”
% 3.2.4 MATLAB™ Analysis & optimization 2D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa

disp('Step 5d) Apply Algorithm: Global search ');
rng default % for reproducibility
fun = @polyline_fitness; % objective
nvars=1;
problem = createOptimProblem('fmincon','objective',fun,'x0',[0]);
% , 'options',...
% optimoptions(@fmincon,'Algorithm','sqp','Display','off');
[x,fval] = fmincon(problem)
gs = GlobalSearch('Display','iter');
%rng(14,'twister') % for reproducibility
[x,fval] = run(gs,problem);
disp(strcat('Global search() minimum found at: t in sec. = ',num2str(x)));
disp(strcat(' signal value =', num2str(fval)));

```

Note*: This MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_polyfit\gs_polyline.m.”

TABLE 3.11 Main program: “Optim_Analysis_3D.m” from MATLAB Analysis and optimization 3D data^a.

```

% by Dr. Jorge Garza-Ulloa

%% Step 1) Data Inspection of Original discrete data
load 3Ddata; %
stem3(x,y,z,'linestyle','none');title(" Step 1a) Plot of Data inspection (Points)")
xlabel("x");ylabel("y");zlabel("z")
% Normalized Data
x=x/max(x);y=y/max(y);z=z/max(z);
stem3(x,y,z,'linestyle','none');title(" Step 1b) Plot of Normalized Data (Points)")
xlabel("x");ylabel("y");zlabel("z")

%% Step 2) Obtain Mathematical model By fitting data
% Fit a Polynomial Surface where x is normalized by mean and std
surffit = fit([x,y],z,'poly23','normalize','on');
% Plot the Fit
figure;
plot(surffit,[x,y],z)
title(" Step 2) Plot of Normalized Data (Points) and Model (Surface)")
xlabel("x");ylabel("y");zlabel("z");

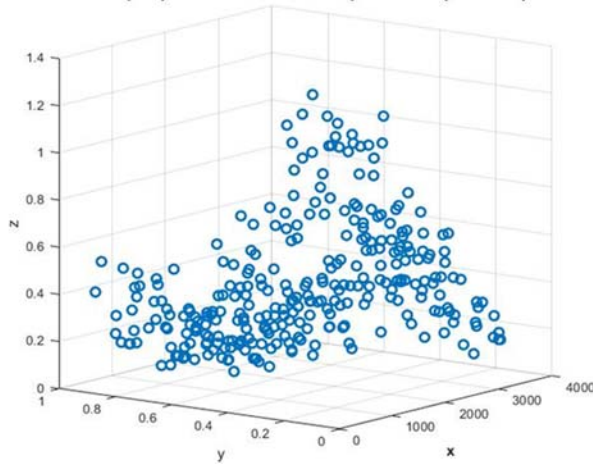
%% Step 3) Evaluate model with residuals
% Plot the residuals surface fit
figure;
plot(surffit,[x,y],z,'Style','Residuals');
title(" Step 3a) Evaluate model with residuals")
xlabel("x");ylabel("y");zlabel("z")
% Plot prediction bounds on the fit.
figure;
plot(surffit,[x,y],z,'Style','predfunc')
title(" Step 3b) Evaluate model for prediction")
xlabel("x");ylabel("y");zlabel("z")
% Evaluate the Fit at a Specified Points specifying a value for x and y ,
% using this form: z = fittedmodel(x,y)
disp(strcat('Evaluation of surface x=.5,y=.5 then z = ',num2str(surffit(0.5,0.5))));

```

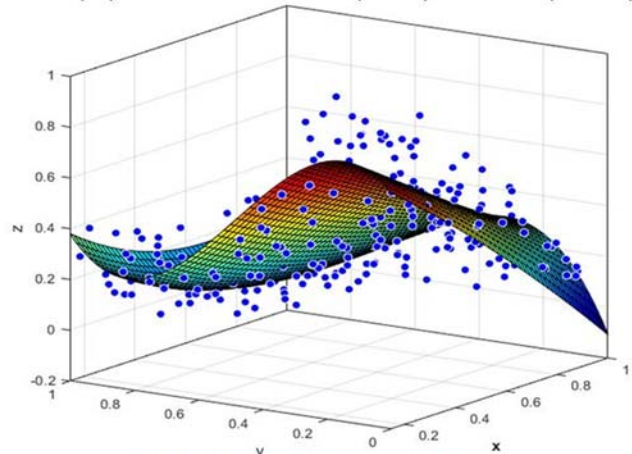
^aThis MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_3dsurface\Optim_Analysis_3D.m.”

3.2.5 MATLAB™ Tutorial Analysis and optimization of 3D data in Biomedical Engineering

Step 1) Plot of Data inspection (Points)



Step 2) Plot of Normalized Data (Points) and Model (Surface)



a) Linear model Poly23:

$$\text{surfFit}(x,y) = p00 + p10*x + p01*y + p20*x^2 + p11*x*y + p02*y^2 + p21*x^2*y + p12*x*y^2 + p03*y^3$$
 where x is normalized by mean 0.5662 and std 0.2494
 and where y is normalized by mean 0.5021 and std 0.2921

b) Coefficients (with 95% confidence bounds):

p00 =	0.3219	(0.2978, 0.346)
p10 =	-0.08175	(-0.1012, -0.06228)
p01 =	-0.33	(-0.3654, -0.2945)
p20 =	0.01678	(0.002221, 0.03134)
p11 =	0.05886	(0.04599, 0.07173)
p02 =	-0.01763	(-0.03217, -0.003093)
p21 =	0.01683	(0.002273, 0.0314)
p12 =	-0.02062	(-0.03515, -0.0061)
p03 =	0.09529	(0.07857, 0.112)

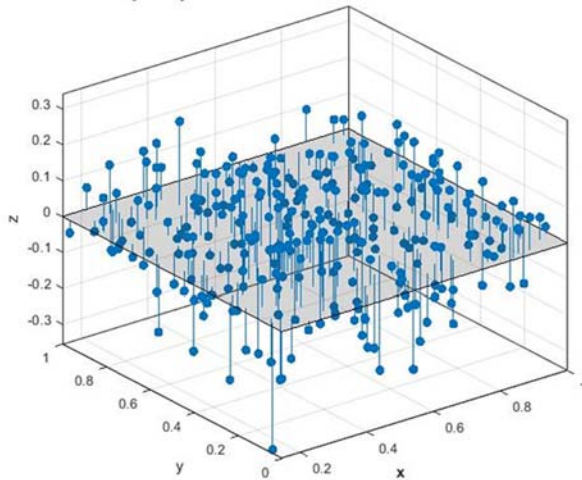
FIGURE 3.11 MATLAB example for the analysis and optimization of “3D” data in biomedical engineering: step 1) Plot of Data inspection points, step 2) Plot of Model obtained by fitting data and original data values: a) Linear Model obtained from the data fitting and b) Coefficients for the linear model.

where “ x ” is normalized by mean and std., and its respective 3D chart is generated, showing the normalized vector values in the axes and the continuous “*surface polynomial function model*” as indicated in the right side of Fig. 3.11. The “*surface linear model poly 3D*” equation is shown at the bottom of Fig. 3.11 a), and their coefficients’ values in Fig. 3.11 b).

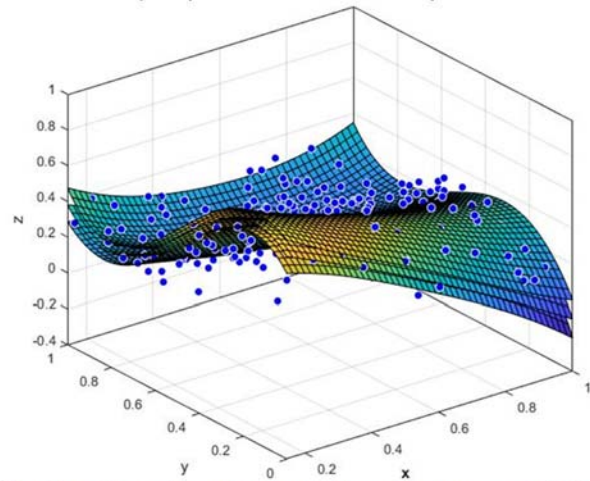
- “Step 3) Evaluate model with residuals” in 3D data, this step is subdivided into three:
 - “Step 3a) Evaluate model with residuals,” which generates a chart showing the residuals values, as indicated in the left-hand side of Fig. 3.12.
 - “Step 3b) Evaluate model for prediction”: a plot prediction bounds of the surface fit is generated as shown on the right-hand side of Fig. 3.12.
 - “Step 3c) Surface Fit evaluated at Specified Point” specifying a value for “ $x = 0.5$ ” and “ $y = 0.5$ ” is calculated to obtain the surface value of “ $z = 0.34728$.”
 - The Function “*surface_fitness.m*” “is listed in Table 3.12. The process is:
 - “Step 4) Define Fitness Function” needed on the “*Evolutionary Algorithms*,” which evaluates the
- “*Fitted Surface Polyline function*” with the parameters, as shown in Eq. (3.6).
- Surface Fitness Functions for 3D optimization**
- $$ft = 0.3219 - 0.08175x - .33y + 0.01678x^2 + 0.05886xy - 0.01763y^2 + 0.01683x^2y - 0.02062xy^2 + 0.09529y^3 \quad (3.6)$$
- Subprogram “*ga_surface.m*” listed in Table 3.13. The process is:
 - “Step 5) Apply Evolutionary Algorithm: Genetic algorithm” defining the parameters needed to call the function “*ga()*” available on “*MATLAB Global Optimization Toolbox*.” It generates the left-hand chart shown in Fig. 3.13.
 - “Step 6) Compare results and select optimal results”: the results are shown in Fig. 3.13a).
 - Subprogram “*ps_surface.m*,” listed in Table 3.14, processes step 5) again, but now applies the “*Particle Swarm Algorithm*”:
 - “Step 5) Apply Evolutionary Algorithm: Particle Swarm Algorithm” defining the parameters needed to call the function “*particleswarm()*” available on “*MATLAB Global Optimization Toolbox*.”
 - “Step 6) Compare results and select optimal results:” it calculates the results, as shown in Fig. 3.13b).

Continue... 3.2.5 MATLAB™ Tutorial Analysis and optimization of 3D data in Biomedical Engineering

Step 3a) Evaluate model with residuals



Step 3b) Evaluate model for prediction



Evaluation of surface $x=.5, y=.5$ then $z = 0.34728$

Step 4) Define a Fitness Function

Surface Fitness Functions =

$$0.3219 - 0.08175x - .33y + 0.01678x^2 + 0.05886xy - 0.01763y^2 + 0.01683x^2y - 0.02062xy^2 + 0.09529y^3$$

FIGURE 3.12 MATLAB example for analysis and optimization of “3D” data in biomedical engineering: step 3a) Plot to evaluate the residual surface model, step 3b) Plot of to evaluate prediction on the surface, and step 4) Definition of the Fitness Function to be used in the Evolutionary Algorithms.

TABLE 3.12 Function: “surface_fitness.m” for MATLAB Analysis and optimization 3D numeric data.

```
% Function “surface_fitness.m”
% 3.3.6 MATLAB™ Analysis & optimization 3D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa

function y = surface_fitness(x)
% Vectorized Surface fitness function
y = 0.3219 - 0.08175*x(:,1) - 0.33*x(:,2) + 0.01678*x(:,1).^2 + ...
    0.05886*x(:,1)*x(:,2) - 0.01763*x(:,2).^2 + 0.01683*x(:,1).^2*x(:,2) - ...
    0.02062*x(:,1)*x(:,2).^2 + 0.09529*x(:,2).^3;
```

Note*: This MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_3dsurface\surface_fitness.m.”

- Subprogram “ds_surface.m,” listed in Table 3.15, process step 5) again, but now applies the “Direct search Algorithm”:
 - “Step 5) Apply Algorithm: Direct search” defining the parameters needed to call the function “patternsearch ()” available on MATLAB Global Optimization Toolbox.” It generates the chart shown on the right-hand side of Fig. 3.13.
 - “Step 6) Compare results and select optimal results”: the results of the calculation are shown in Fig. 3.13c).
- The subprogram “gs_surface.m,” listed in Table 3.16, processes step 5) again, but now applies the “Global search Algorithm”:
 - “Step 5) Apply Algorithm: Global search” defining the parameters needed to call the function “globalsearch ()” available on MATLAB Global Optimization Toolbox.”
 - Step 6) Compare results and select optimal results”: it calculates the results as shown in Fig. 3.13 d).

TABLE 3.13 Subprogram to use “Genetic Algorithm: “ga_surface.m” for MATLAB Analysis and optimization 3D numeric data^a.

```

Sub-program "ga_surface.m"
% 3.2.5 MATLAB™ Analysis & optimization 3D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
disp('Step 5a) Apply Evolutionary Algorithms: Genetic Algorithm');
rng default % For reproducibility
FitnessFunction = @surface_fitness;
numberOfVariables = 2;
lb = [0,0];%warning('off','globaloptim:checkbound:lengthOfUpperBound');
ub = [1,1];%warning('off','globaloptim:checkbound:lengthOfLowerBound')
options = optimoptions('ga'); % Default options
% 10 Generations of 100 chromosomes
options = optimoptions(options,'MaxGenerations', 10);
options = optimoptions(options,'MaxStallGenerations', inf);
options = optimoptions(options,'FunctionTolerance', 0);
%options = optimoptions(options,'ConstraintTolerance', 0);
options = optimoptions(options,'Display', 'off');
options = optimoptions(options,'PlotFcn', { @gaplotbestf });
[x,fval] = ga(FitnessFunction,numberOfVariables,[],[],[],[],lb,ub,[],options);
disp('Genetic Algorithm');
disp(strcat('Global minimum found x= ',num2str(x(1)), ' y= ',num2str(x(2))));
disp(strcat(' z= ', num2str(fval)));
    
```

^aThis MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_3dsurface\ga_surface.m.”

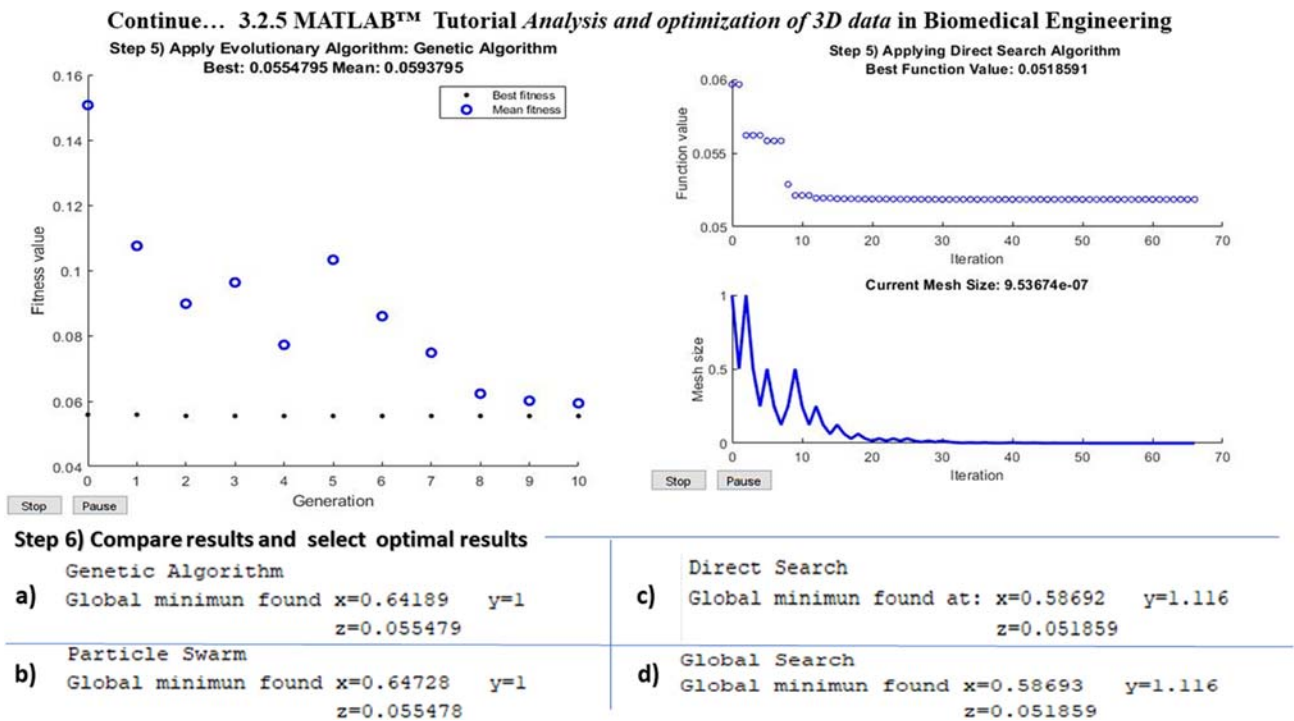


FIGURE 3.13 MATLAB example for analysis and optimization of “3D” data in Biomedical engineering: step 5) Genetic Algorithm results of Fitness Function on 10 generations, step 5) Direct Search Algorithm results of Fitness Function on 70 iterations: a) Genetic Algorithm optimization result, b) Particle Swarm Algorithm optimization result, c) Direct Search algorithm optimization results, and d) Global Search optimization results.

TABLE 3.14 Subprogram to use “Particle Swarm Algorithm”: “ps_surface.m” from Section 3.3.6 MATLAB Analysis and optimization 3D numeric data^a.

```

% Function “ps_surface.m”
% 3.2.5 MATLAB Analysis & optimization 3D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
disp(“Step 5b) Apply Evolutionary Algorithms: Particle Swarm”);
rng default % for reproducibility
fun = @surface_fitness;% objective
nvars = 2;
lb = [0,0];%warning(“off”,“globaloptim:checkbound:lengthOfUpperBound”);
ub = [1,1];%warning(“off”,“globaloptim:checkbound:lengthOfLowerBound”);
options = optimoptions(“particleswarm”,“SwarmSize”,1000);
[x,fval,exitflag] = particleswarm(fun,nvars,lb,ub,options);
disp(“Particle Swarm”);
disp(strcat(“Global minimum found x =”,num2str(x(1)),“y =”,num2str(x(2))));
disp(strcat(“z =”, num2str(fval)));

```

^aThis MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_3dsurface\ps_surface.m.”

TABLE 3.15 Subprogram to use “Direct search Algorithm”: “ds_surface.m” from Section 3.3.6 MATLAB Analysis and optimization 3D numeric data.

```

% Function “ds_surface.m”
% 3.2.5 MATLAB Analysis & optimization 3D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
disp(“Step 5c) Apply Evolutionary Algorithms: Direct search”);
options = optimoptions(“patternsearch”,“PlotFcn”,{@psplotbestf,@psplotmeshsize});
[x,fval] = patternsearch(@surface_fitness,[1 1],[],[],[],[],[],[],[],options);
disp(“Direct Search”);
disp(strcat(“Global minimum found at: x =”,num2str(x(1)),“y =”,num2str(x(2))));
disp(strcat(“z =”, num2str(fval)));

```

Note*: This MATLAB program can be downloaded from the website companion and installed in the following path “. . .\Exercises_book_ABME\CH3\GA_Matlab\GA_3dsurface\ds_surface.m.”

Conclusion

Four methods were used to calculate the “3D optimization” requested in this problem to find the “global minimum value,” as summarized at the bottom of Fig. 3.13; two of them used “Evolutionary Algorithms: Genetic Algorithm and Particle Swarm Algorithm” to produce very similar results of: “Minimum found at $x = 0.64$, $y = 1$ with a $z = 0.054$,” and two other “Traditional algorithms for search: Direct search and Global search” with optimization showing the following results: “Minimum found at $x = 0.586$, $y = 1.116$ with a $z = 0.0518$.” Because the value is evaluated at “ $y > 1$,” then we can be sure that “the result obtained in the two Evolutionary Algorithms of $z = 0.054$ is the global minimum and is inside of the range “ $y \leq 1$.”

3.4 IBM Watson Studio for artificial intelligence

In Chapter 2, Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to help in the Solution of AI Biomedical Engineering Problems, we introduced “IBM Cloud” as a robust suite of advanced data and “AI tools,” including an “Application Program Interface (API)” that defines a set of routines, protocols, and tools for using or building software applications. We created services using “IBM Cloud “solution for Natural Language Processing,” such as “Speech to Text,” “Text to Speech,” “Natural Language Understanding,” and others. In this chapter the main objective is to study “IBM Watson Studio solutions for Artificial Intelligence.”

TABLE 3.16 Subprogram to use “Global search Algorithm”: “gs_surface.m” from Section 3.3.6 MATLAB Analysis and optimization 3D numeric data.

```

% Function “gs_surface.m”
% 3.2.5 MATLAB™ Analysis and optimization 3D data in Biomedical Engineering
% Book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
% by Dr. Jorge Garza-Ulloa
disp(“Step 5d) Apply Evolutionary Algorithms: Global Search”)
rng default % for reproducibility
fun = @surface_fitness; % objective
nvars = 2;
x0 = [1,1];
problem = createOptimProblem(“fmincon”,“objective”,fun,“x0”,[0 0]);
%, “options”,. . .
% optimoptions(@fmincon,“Algorithm”,“sqp”,“Display”,“off”);
[x,fval] = fmincon(problem)
gs = GlobalSearch(“Display”,“iter”);
rng(14,“twister”) % for reproducibility
[x,fval] = run(gs,problem);
disp(“Global Search”);
disp(strcat(“Global minimum found x =”,num2str(x(1)),“y =”,num2str(x(2))));
disp(strcat(“z =”, num2str(fval)));

```

Note*: This MATLAB program can be downloaded from the website companion and installed in the following path “. . . \Exercises_book_ABME\CH3 \GA_Matlab\GA_3dsurface\gs_surface.m.”

“IBM Watson Studio” is a collaborative environment with graphical tools for designing, training, deploying, and managing models with “Watson Machine Learning services,” such as “SPSS modeler,” “AutoAI,” “Neural network modeler,” “Notebooks,” “Experiment builder,” “Decision Optimization model,” “Spark MLlib,” and other “AI tools,” where:

- “SPSS modeler” presents a graphical view of your model while you build it by combining nodes representing objects or actions.
- “AutoAI” experiments automatically preprocess your data, select the best estimator for the data, and then generate model candidate pipelines for you to review and compare. To “deploy the best performing pipeline as a machine learning model.”
- “Neural network modeler” presents a graphical view of your model while you build it by combining “neural network nodes.”
- “Notebooks” provides an interactive programming environment for working with data, testing models, and rapid prototyping.
- “Experiment builder” automates the running hundreds of training runs while tracking and storing results.
- “Decision Optimization model” guides you through building and solving prescriptive models.
- “Spark MLlib modeler” presents a graphical view of your model while you build it by combining nodes representing algorithm nodes.

“IBM Watson Studio” has many key capabilities for the development of “AI systems and applications,” such as:

- Simplified steps to prepare, blend, and analyze data,
- Easy to use visualization,
- Drag and drop “Machine Learning (ML)” with “SPSS Modeler,”
- Open source integration,
- Offline processing while keeping data on the desktop,
- Reuse of skills across “Watson Studio developments tools,” and
- And many more.

Actually, “IBM Watson Studio” as an AI platform can be used in three ways: “Watson Studio Cloud,” “Watson Studio Local,” and “Watson Studio Desktop.” Their capabilities, typical users, and license type are indicated in Table 3.17.

The “General Typical Steps in Data Science to build and develop AI Applications” are: “Prepare and Visualize data,” “Build and Validate AI Models,” and “Deploy and Optimize AI Models,” as indicated in Fig. 3.14.

This chapter will focus on “Step 1) Read, Prepare and Visualize data “using the IBM Watson Studio for Artificial Intelligence Tool IBM SPSS Modeler Flow.” The other chapters will focus on “Step 2) Build and Validate AI Models”

(Continued)

TABLE 3.17 Watson Studio as an AI platform can be used in three ways: “Watson Studio Cloud,” “Watson Studio Local,” and “Watson Studio Desktop.”

Platform	Capabilities	Users	License type
Watson Studio Cloud	Data Science team collaboration on a public cloud	Organizations with multiple data scientists and data science teams	SaaS (Software as a service)
Watson Studio Local	Data Science team collaboration behind a firewall	Organizations with multiple data scientists and data science teams	Perpetual—Term pricing monthly
Watson Studio Desktop	Data Science individual starting with SPSS Modeler canvas, data shaping, projects and notebook	Out of the box enterprise solution for data scientists and data engineers. A suite of data science tools, such as RStudio, Spark, Jupyter, and Zeppelin notebook	Licensed programs Watson Studio Local counts the daily user license usage by Authorized Users. Users with any of the following privileges count toward this license: user or administrator.

Note: All the IBM Watson Studio example, tutorials and exercise in this book are using “Watson Studio Cloud.”

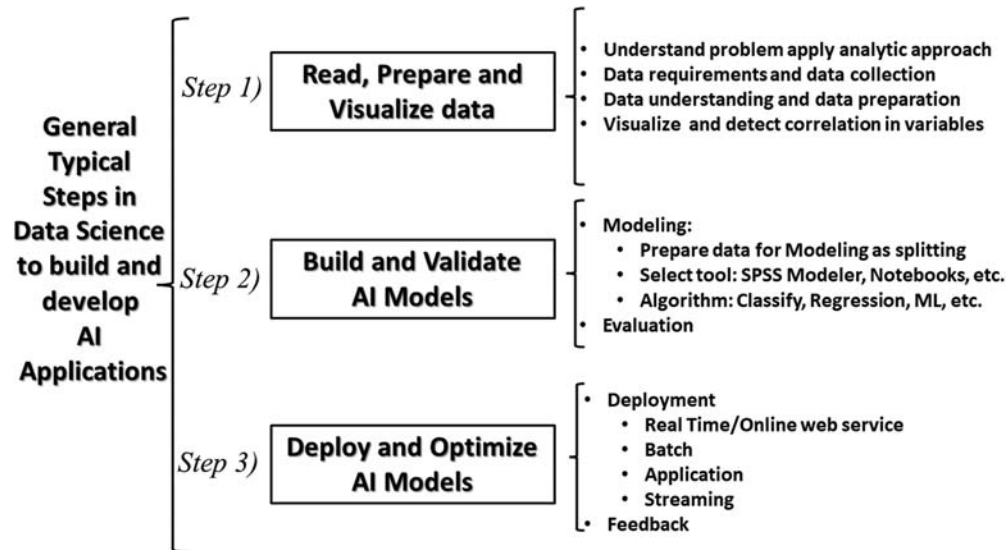


FIGURE 3.14 General typical steps in Data Science to build and develop AI applications.

(Continued)

and part of “Step 3) Deploy and Optimize AI Models” using: “Machine Learning Models.” Additionally, we will cover other “Watson Machine Learning services” in the rest of the book.

3.4.1 IBM SPSS Modeler Flow

The “SPSS (Statistical Package for the Social Sciences) Modeler flows” from “IBM Watson Studio” is an excellent group of “AI tools” to develop predictive models and deploy them to improve decision-making. The flow interface supports the data mining visual modeling

process integrating algorithms from “AI,” “ML,” “CC,” and “statistics.” It has many methods available on the “node palette” using the “flow editor” that allow one to derive new information from the data to develop predictive models.

“SPSS Modeler flows” has the following capabilities: “read, prepare, and visualize data,” “Build and Validate AI Models,” and “Deploy and Optimize AI Models.” Each capability can be summarized as follow:

- “Read, prepare, and visualize data”
 - Read any data size formatted as relational (tables in relational data sources), tabular (Excel files *.xls or CSV files) and textual (in supported relational tables or files).

- Prepare data using automatic data preparations functions and applying SQL statements, as well as cleanse, shape, sample, sort, and derive data.
- Visualize data with many chart options, such as charts, plot, multiplot, time plot, distribution, collection, and others. Also allows identify natural language from text fields.
- **“Build and Validate AI Models”**
 - Build predictive models using automatic modeling functions or choose from many modeling algorithms. Also classify textual data and identify relationships between concepts in textual data.
- **“Deploy and Optimize AI Models”**
 - Output nodes provide the means to obtain information about the data and models.
 - Export nodes provide a mechanism for exporting data in various formats to interface with other software tools.

“SPSS Modeler flows” is organized to use the following “nodes palettes”: “import,” “record operations,” “field operations,” “graphs,” “modeling,” “outputs,” and “exports,” as shown in Fig. 3.15*.

Note*: “Modeling,” “Outputs,” and “Export” are going to be explained in the next chapters.

- **“Import”**: allows import data to be stored in various formats, or to generate your own synthetic data
- **“Record Operations”**: allows making changes to data at record level for data preparation.

- **“Field Operations”**: allows to select, clean and construct data for the preparation analysis
- **“Graphs”**: allows the data mining visualization process.
- **“Modeling”**: allows the integration of a big variety of AI modeling methods and statistics.
- **“Text Analytics”**: allows to identify languages, text link analysis, text mining, and others text features.
- **“Outputs”**: allows to obtain information form the modeling methods.
- **“Export”**: allows the exporting of data in various formats.

The Menu Nodes from “IBM Watson Studio—SPSS Modeler Flower” related with “Step 1) Read, Prepare and Visualize data” are shown in Fig. 3.15 with their general description. The other “Menu Nodes” are going to be explained in the next chapters *chapter 4 Machine learning models applied to Biomedical Engineering, chapter 5 Deep Learning Models Principles applied to Biomedical Engineering and chapter 6 Deep Learning Models Evolution Applied to Biomedical Engineering*

“SPSS Modeler flows” is very useful for developing many biomedical engineering applications. It has the following capabilities: “read, prepare, and visualize data,” “build and validate AI models,” and “deploy and optimize AI models.”

IBM Watson Studio - SPSS Modeler Flower Menu Nodes for Step 1) Read, Prepare and Visualize data

Import	Field Operations	Graphs
Data Asset: Import data from a data access	Auto Data Prep: Automatically prepares data for modelling	Chart: Chart builder /create chart def.
User Input: Input data to test dataset	Type: Define field metadata and purpose	Plot: For relationship in num. fields
Slim Gen: Generate simulated data	Filter: Specify filter settings	Multiplot: Multiple Y over one X field
	Derive: Modify data values /derive new fields from data	Time Plot: View one or more time series
	Filler: Replace field values and change storage	Distribution: Occurrence of non num.
	Reclassify: For transformation of categorical values	Histogram: Occurrence of numeric data
	Binning: Automatically create new nominal fields from fields	Collection: Distribution field to others
	RFM Analysis: Recency, Frequency, Monetary (RFM) Analysis	Web: Relationships between values
	Ensemble: Combines two or more model for better predictions	Evaluation: Evaluate/compare models
	Partition: Splits data into separate subsets	
	SetToFlag: Derive flag fields based on the categorical values	
	Restructure: Multiple fields for nominal / flag field	
	Transpose: Swap the data in rows and columns	
	Field Reorder: Define natural order for disp. fields downstream	
	History: Used for sequential data, such as time series data	
	Time Intervals: Specify intervals and derive a new time field	
	Anonymize: Disguise field names, field values for avoid exposure	
	Reproject: Change projected coordinates to geographic systems	

Note: The following “Menu Nodes” are going to be explained in the next chapters

- Modeling
- Text Analytics
- Outputs
- Export

FIGURE 3.15 SPSS Modeler Flower Menu Nodes for Step 1) Read, Prepare, and Visualize data.

3.4.2 IBM Watson using SPSS Modeler Flow for general dataset analysis

3.4.2.1 Research 35 IBM Watson using SPSS Modeler Flow for “diabetes” analysis

3.4.2.1.1 General objective

“Use “IBM Watson Studio IBM SPSS Modeler Flow” for a basic analysis of a “diabetes” dataset with the objective “to find relations between their variables.”

3.4.2.1.2 Specific objectives

1. “Create an IBM Cloud service for Watson Studio,” that allows the creation of custom models to apply embedding of “Artificial Intelligence (AI) tools” and “Machine Learning (ML)” algorithms.
2. “Create a new project for AI and Cognitive Models using an IBM Cloud Storage service” to add data, refine data, and specify “AI”–“ML” assets types needed in this project.
3. “Create an asset type for Modeler Flow type and IBM SPSS Modeler”
4. “Design a Model Flow to analyze a diabetes dataset” generating:
 - a. “Graph distribution” for the “class” * field that can be either: “tested_negative” or “tested_positive.”
 - b. “3D graph” to visualize the relation between the fields: “age” *, “pedi * = Diabetes pedigree function” and “class” *.
 - c. “Multiplot” to visualize the relation between the fields: “plas * = plasma glucose concentration” and “class*” versus the other fields.

Note*: See Table 3.18 for see the fields of the diabetes dataset.

5. Make conclusions about the analysis of the “diabetes dataset establishing the relationship between all fields in this specific data file.”

Note*: In this research tutorial of “IBM SPSS Modeler Flow,” we concentrate on the use and explanation of the “menu nodes pallets”: “import,” “record operations” “field operations” and “graphs.” The other “menu nodes pallets” such as “modeling,” “text analytics,” “outputs,” and “export” will be explained in the Chapter 4, Machine Learning Models Applied to Biomedical Engineering.”

3.4.2.1.3 Dataset

The dataset “diabetes.csv” is based on information at the “National Institute of Diabetes and Digestive and Kidney Diseases with the title: Pima Indians Diabetes Database.” It has: “768 records,” with “eight fields (attributes),” as indicated in Table 3.18.

3.4.2.1.4 Procedure

The steps to “analyze a diabetes dataset to find relations between their variables by applying IBM SPSS Modeler Flow” are summarized in Table of slides 3.1 and each step of the example is visually explained using screen sequences with easy to follow figures.

Note: The “IBM Cloud website is evolving in an exponential way every day,” some screens could be updated. I recommend to understand very well the objectives, apply them accordingly with the new screens’ formats and the current “IBM Cloud website contents.”

TABLE 3.18 Dataset “diabetes.csv” fields and descriptions.

Field	Description ^a 768 instances of patients. All females >21 and <81 years old
preg	Number of times pregnant (integers)
plas	Plasma glucose concentration 2-h oral glucose tolerance test
pres	Diastolic blood pressure in mm Hg (real number)
skin	Triceps skin fold thickness in mm (real number)
insu	2-h serum insulin in mu U/mL (real number)
mass	Body mass index based on weight in kg/(height in m) ² (real number)
pedi	Diabetes pedigree function (real number). It is a function which scores likelihood of diabetes based on family history
age	Age in years (integers)
class	Class variable with two string: ['tested_negative', 'tested_positive']

^aThis dataset can be downloaded from the website companion and installed in the following path “. . .Exercises_book_ABME\CH3\WS\IBM\IBM_SPSS\diabetes.csv.”

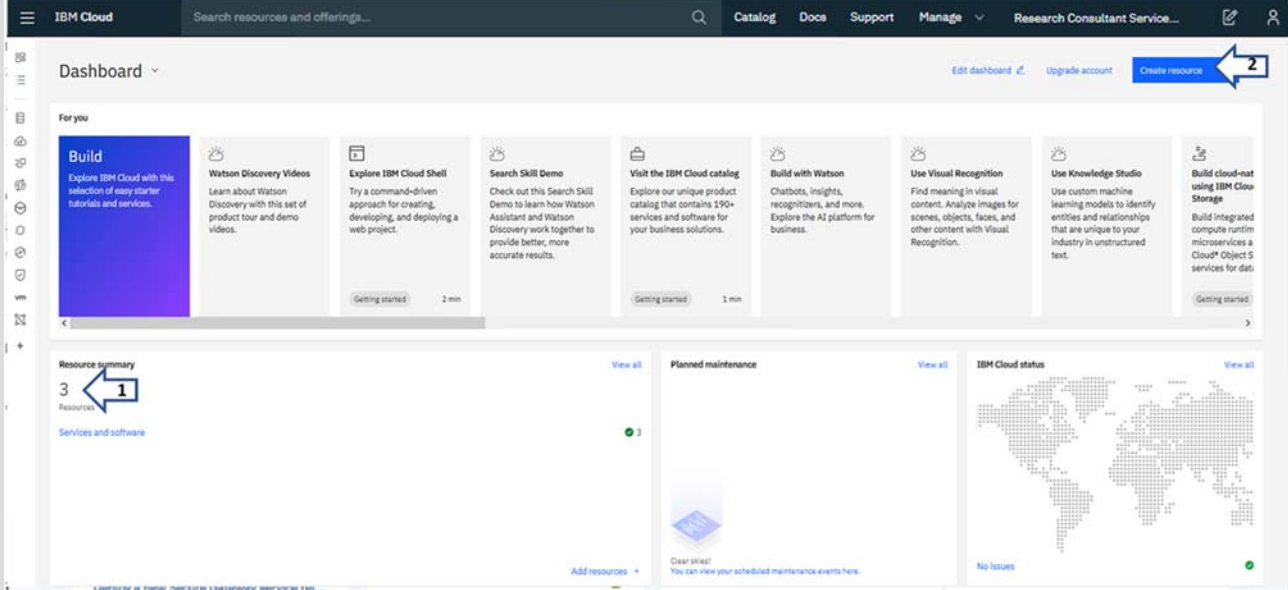
Table of slides 3.1 Steps for the example “Basic analysis of a diabetes dataset to find relations between their variables by applying IBM SPSS Modeler Flow.”

Slide Description

1 Login to your IBM Cloud Account at the website: <https://cloud.ibm.com/login> entering your “IBMid and Password”
Note: You must have the three services available that were created in Chapter 2 examples, then click the “Create resource” button

Screen figure

1. Login in your account IBM Cloud Account in the website: <https://cloud.ibm.com/login> entering your IBMid and Password



The screenshot shows the IBM Cloud dashboard. At the top right, there is a 'Create resource' button with a blue arrow pointing to it labeled '2'. Below the dashboard, there is a 'Resource summary' section with a blue arrow pointing to it labeled '1'. The 'Resource summary' section shows '3 Resources' and 'Services and software'. Below the 'Resource summary' section, there is a text box that says 'You must have the 3 services available created on Chapter 2 examples, then click the “Create resource” button'. At the bottom of the screenshot, there is a text box that says 'From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa'.

You must have the 3 services available created on Chapter 2 examples, then click the “Create resource” button

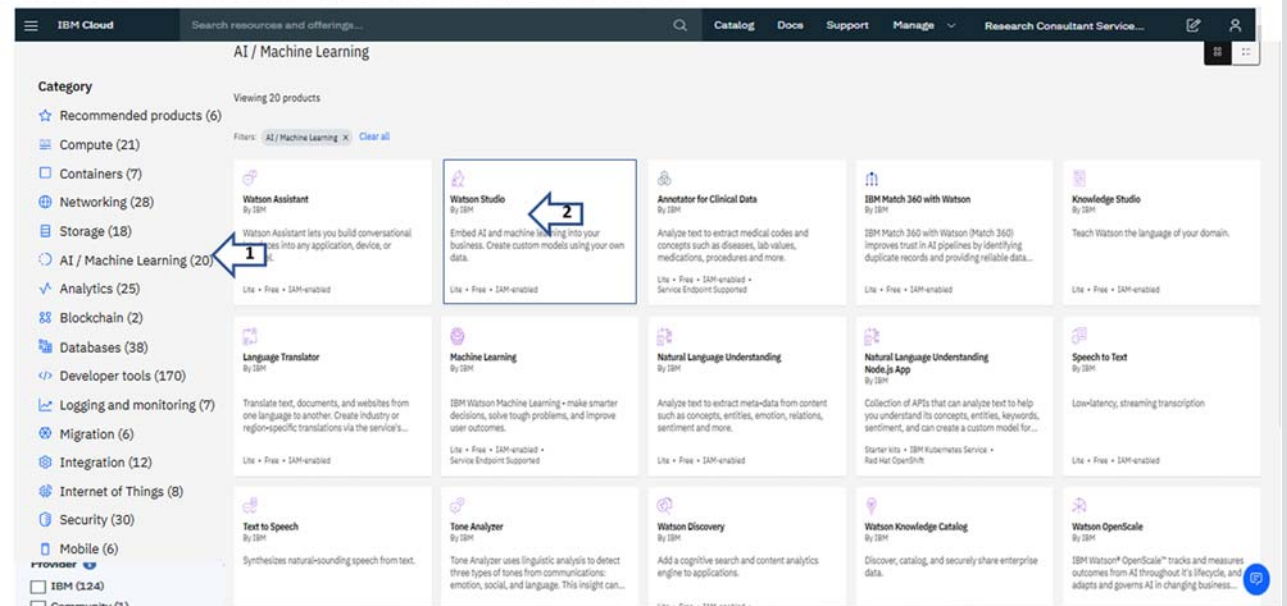
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 2 Description In the IBM Cloud screen on “AI/Machine Learning Catalog services” Note: Click on “AI/ Machine Learning, then click on Watson Studio” to create the service as indicated in the slide with “2”

Screen figure 2. In the IBM Cloud screen on “AI /Machine Learning Catalog services”



Click on “AI / Machine Learning”, then click on “Watson Studio” to create the service as indicated in the slide with “2”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

3 In the IBM Cloud dashboard screen, create the free “Watson service”
Note: Select a location near to you, click on license agreement, and press the button “Create”

3. In the IBM Cloud dashboard screen, create the free “Watson service”

The screenshot shows the IBM Cloud dashboard for creating a Watson Studio service. The interface is divided into three main sections:

- Left Sidebar:** Contains service details such as 'Type: Service', 'Provider: IBM', 'Updated on: 09/24/2021', 'Category: AI / Machine Learning', and 'Compliance: HIPAA Enabled, SAP-enabled'. It also lists 'Location' options: Frankfurt, London, Tokyo, and Dallas.
- Main Content Area:**
 - Select a location:** A dropdown menu is set to 'Dallas (us-south)', with a blue arrow labeled '1' pointing to it.
 - Select a pricing plan:** A table compares two plans: 'Lite' (Free) and 'Standard v1' (\$99.00 USD/Instance). The 'Lite' plan includes 1 authorized user, 50 capacity units, and 4 environments. The 'Standard v1' plan includes 1 authorized user, unlimited viewer collaborators, and 50 capacity units.
- Right Panel:** A 'Summary' section shows 'Watson Studio' as a 'Free' service. At the bottom, there is a checkbox for 'I have read and agree to the following license agreements:' with a 'Terms' link, and a blue 'Create' button with a blue arrow labeled '3' pointing to it.

Select a location near to you, click on license agreement, and press the button “Create”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide

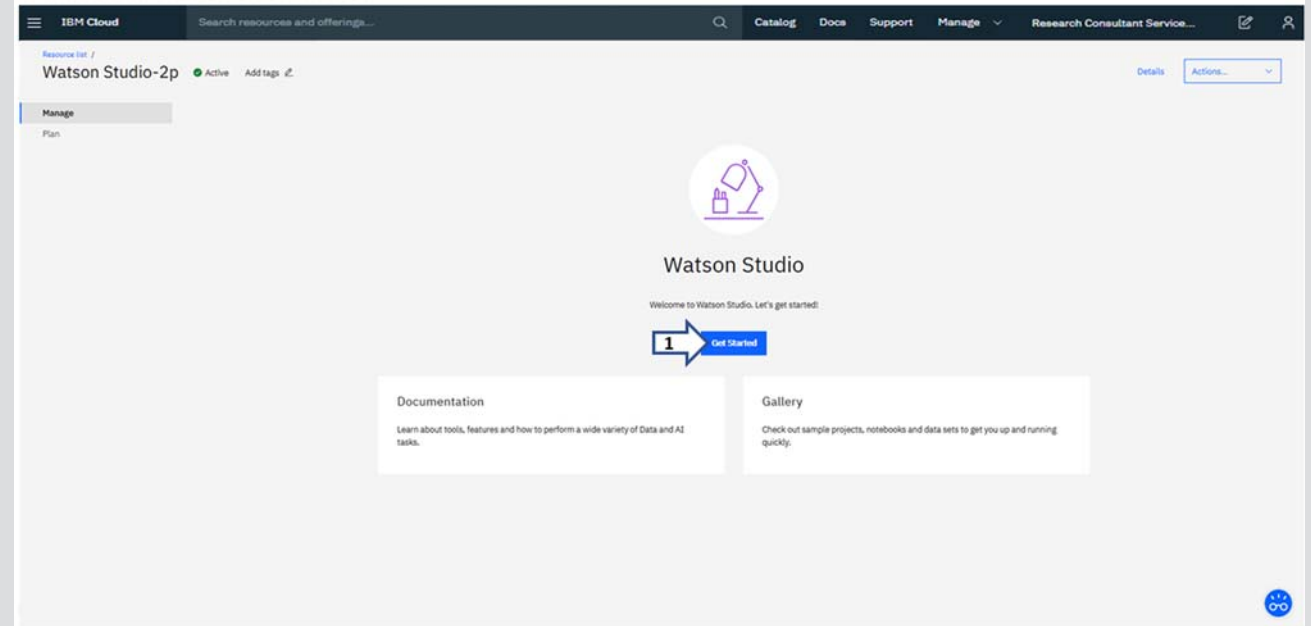
Description

Screen figure

4

In the IBM Watson Studio screen
Note: Press the button "Get Started"

4. In the IBM Watson Studio screen



Press the button "Get Started"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 5 In the IBM Watson Studio provide your information
Note: Enter your company name, phone number, select how do you want to be contacted, and click “Continue”. Until appears the button “Go to IBM Watson Studio”, and click on it.

5. In the IBM Watson Studio provide your information

The screenshot shows the IBM Watson Studio registration page. At the top, there is a navigation bar with the IBM Watson Studio logo, an 'Upgrade' button, a notification bell, and a user profile dropdown labeled 'Research Consultant Service...'. The main heading is 'Provide your information to continue'. Below this, there are four numbered callouts: 1 points to the 'Company name' input field containing 'RESEARCH CONSULTANT SERVICES'; 2 points to the 'Phone number' input field containing '+1' and '9159999999'; 3 points to the 'by email' radio button, which is selected; and 4 points to the 'Continue' button. To the right of the form, there is a consent section: 'IBM may use my contact data to keep me informed of products, services, and offerings:' followed by 'by email' (checked) and 'by phone' (unchecked). Below this is a paragraph: 'You can withdraw your marketing consent at any time by submitting an opt-out request. Also, you may unsubscribe from receiving marketing emails by clicking the unsubscribe link in each email. More information on our processing can be found in the IBM Privacy Statement.' and another paragraph: 'By submitting this form, I acknowledge that I have read and understand the IBM Privacy Statement and I accept the product Terms and Conditions of this registration form.' At the bottom, a blue progress bar is shown with the text 'Done! Your IBM Watson Studio apps are ready to use.' and a 'Go to IBM Watson Studio' button.

Enter your company name , phone number, select how do you want to be contacted, and click “Continue”. Until appears the button “Go to IBM Watson Studio” and click on it.

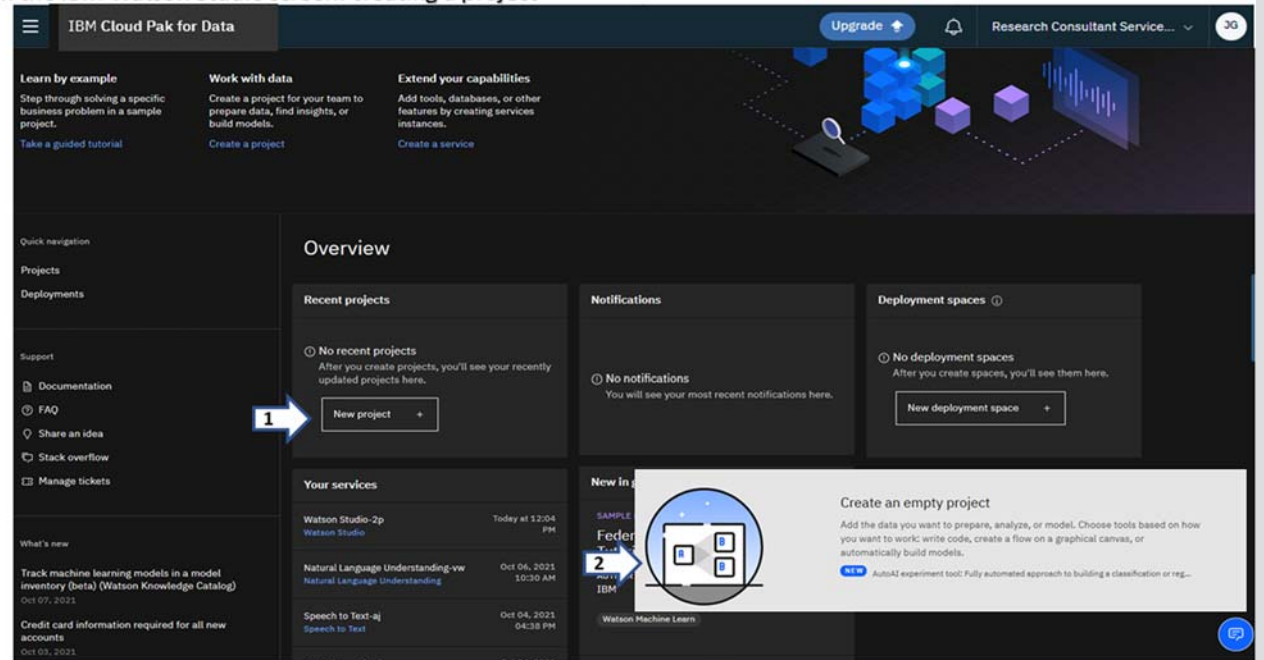
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 6 Description
In the IBM Watson Studio screen: creating a project
Note: Watch the demo and finally select the button "New project", then click on the option "Create an empty project"

Screen figure
6. In the IBM Watson Studio screen: creating a project

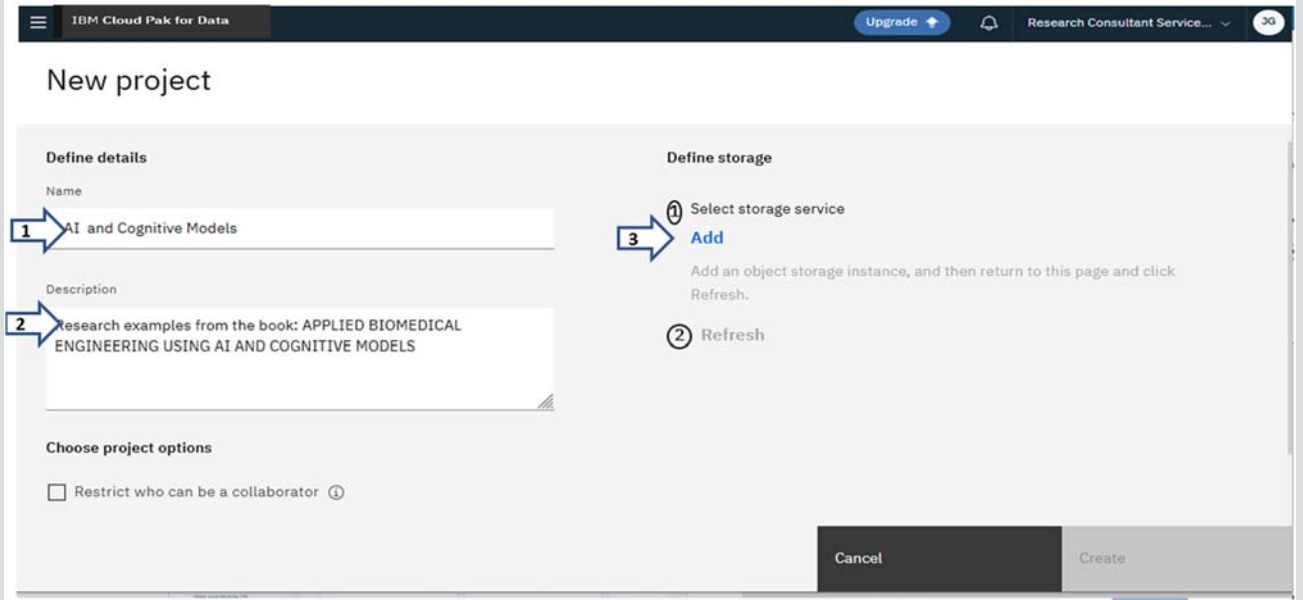


Watch the demo and finally select the button "New project", then click on the option "Create an empty project"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 7 In the IBM Watson Studio—new project screen, enter the project name and description
Note: In Define storage: “1 Select storage service” and click “Add”

7. In the IBM Watson Studio - new project screen, enter the project name and description



In Define storage selection “1 Select storage service” and click “Add”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

8 In the IBM Watson Studio “Cloud Object Storage”: create a new, service pressing the button “Create”
Note: Take note of your assigned Service name

8. In the IBM Watson Studio “Cloud Object Storage”: create a new, service pressing the button “Create”

The screenshot displays the IBM Cloud Pak for Data interface. At the top, there's a navigation bar with 'IBM Cloud Pak for Data', an 'Upgrade' button, and a user profile 'Research Consultant Service...'. Below this is the 'Services catalog' section for 'Cloud Object Storage'. The page includes a 'Create' button (highlighted with a blue arrow labeled '2') and an 'About' button. A summary panel on the right shows the following details:

- Summary**
- Cloud Object Storage**
- Region: Global
- Plan: Lite
- Service name: Cloud Object Storage-ac
- Resource group: Default

The 'Pricing plan' section indicates that displayed prices do not include tax and are for the United States. A table lists the 'Lite' plan with the following features:

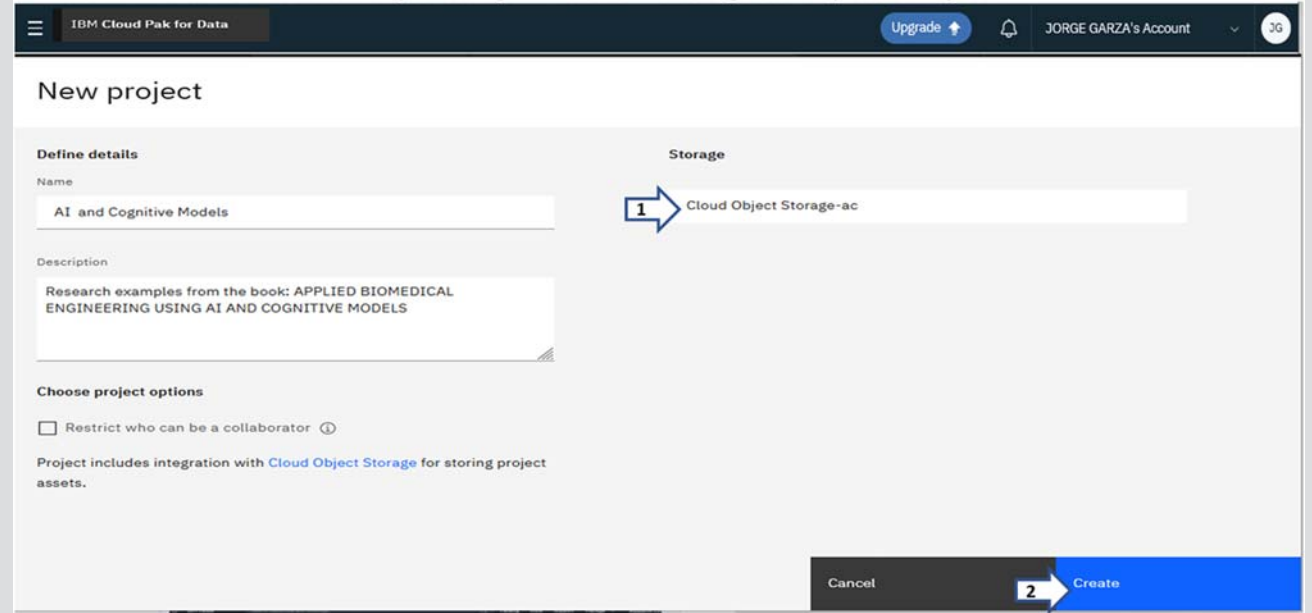
Plan	Features	Pricing
Lite	1 COS Service Instance Storage up to 25 GB/month Up to 2,000 Class A (PUT, COPY, POST, and LIST) requests per month Up to 20,000 Class B (GET and all others) requests per month Up to 10 GB/month of Data Retrieval Up to 5GB of egress (Public Outbound) Applies to aggregate total across all storage bucket classes The Lite service plan for Cloud Object Storage includes Regional and Cross Regional resiliency, flexible data classes, and built in security. Lite plan services are deleted after 30 days of inactivity.	Free

Take note of your assigned Service name

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

9 In the IBM Watson Studio “Cloud Object Storage”: in the define storage section press “2. Refresh”
Note: in IBM Watson Studio “New project,” verify that the “Storage” service is available and press the button “Create”

9. In the IBM Watson Studio “Cloud Object Storage”: in the define storage section press “2. Refresh” .



In IBM Watson Studio “New project”, verify that the “Storage” service is available and press the button “Create”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

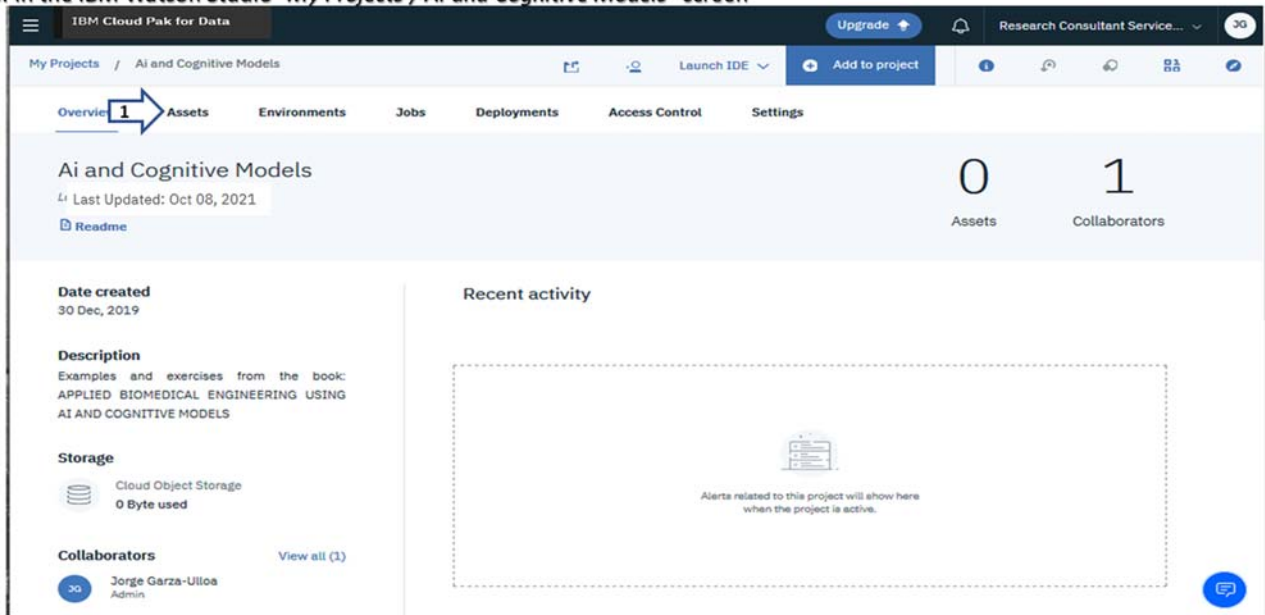
(Continued)

Slide Description

10 In the IBM Watson Studio "My Projects / AI and Cognitive Models" screen
Note: Select "Assets" with a click

Screen figure

10. In the IBM Watson Studio "My Projects / AI and Cognitive Models" screen



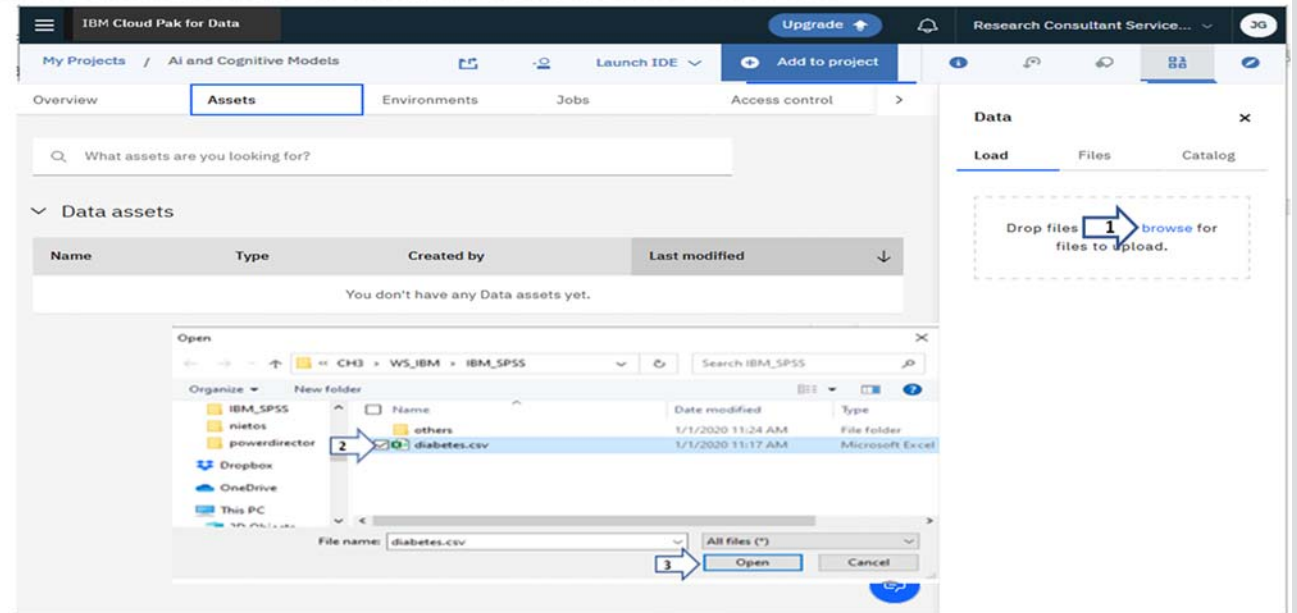
The screenshot shows the IBM Watson Studio interface for a project named "AI and Cognitive Models". The navigation bar at the top includes "Overview", "Assets", "Environments", "Jobs", "Deployments", "Access Control", and "Settings". A red arrow points to the "Assets" tab. The main content area displays the project name, last updated date (Oct 08, 2021), and a "Readme" link. On the right, there are two large numbers: "0 Assets" and "1 Collaborators". Below this, there are sections for "Date created" (30 Dec, 2019), "Description" (Examples and exercises from the book: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS), "Storage" (Cloud Object Storage, 0 Byte used), and "Collaborators" (Jorge Garza-Ulloa, Admin). A "Recent activity" section is also visible, showing a placeholder for alerts. At the bottom, there is a text box that reads "From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa".

Select "Assets" with a click

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

11 In the IBM Watson Studio "My Projects / AI and Cognitive Models" section "Assets"
Note: Load the assets using "browse," select the dataset "diabetes.csv" in the data companion directory and press the button "Open"

11. In the IBM Watson Studio "My Projects / AI and Cognitive Models" section "Assets"



Load the assets using "browse", select the dataset "diabetes.csv" in the data companion directory and press the button "Open"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

12 In IBM Watson Studio "My Projects / AI and Cognitive Models" section Assets: observe the loaded asset "diabetes" available
Note: Click on the button "Add to project"

Screen figure

12. In IBM Watson Studio "My Projects / AI and Cognitive Models" section Assets: observe the loaded asset "diabetes" available

The screenshot shows the IBM Watson Studio interface. At the top, there is a navigation bar with 'IBM Cloud Pak for Data' and 'Upgrade' buttons. Below that, the breadcrumb 'My Projects / AI and Cognitive Models' is visible. The main navigation tabs include 'Overview', 'Assets', 'Environments', 'Jobs', 'Deployments', 'Access Control', 'Load', 'Files', and 'Catalog'. The 'Assets' tab is active, showing a search bar and a 'Data assets' section with '0 asset selected.' Below this is a table with columns: NAME, TYPE, CREATED BY, LAST MODIFIED, and ACTIONS. A single row is visible with the following data: checkbox, CSV, diabetes.csv, Data Asset, Jorge Garza-Ulloa, Oct 08, 2021, 12:40 PM. A red arrow labeled '1' points to the checkbox. Another red arrow labeled '2' points to the 'Add to project' button in the top navigation bar. On the right side, there is a 'Load' panel with instructions: 'Drop files here or browse for files to upload.' and 'Stay on the page until upload completes. Incomplete uploads are cancelled.'

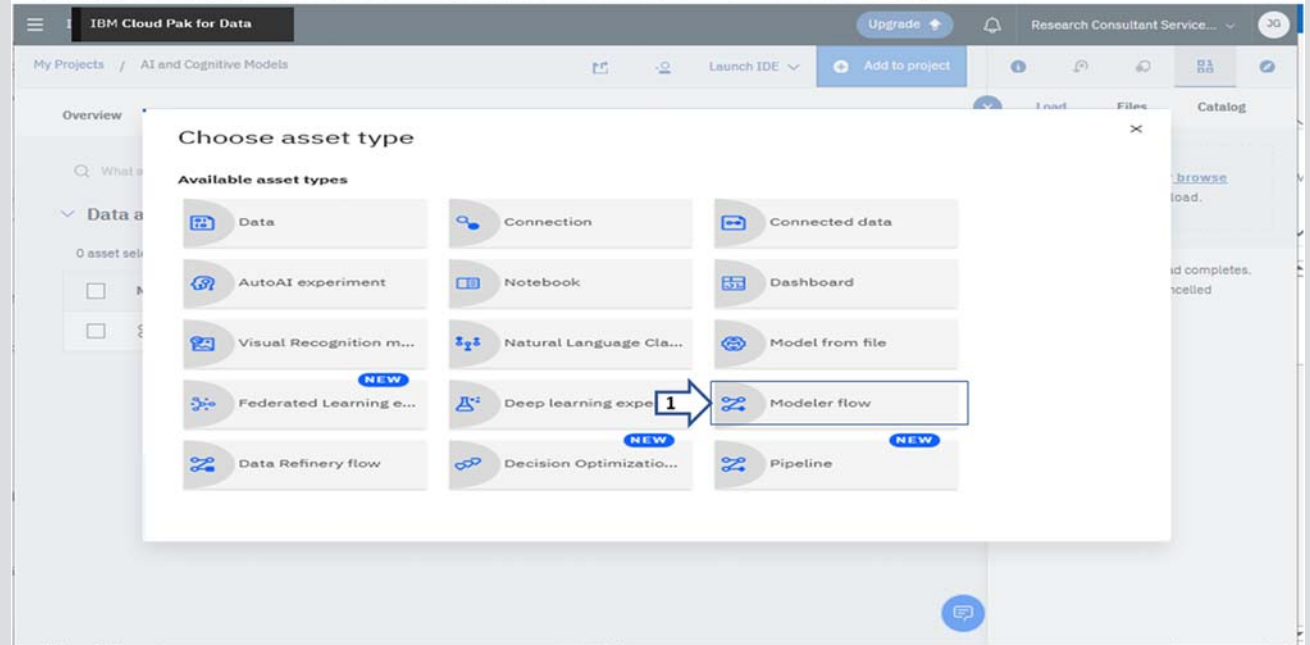
	NAME	TYPE	CREATED BY	LAST MODIFIED	ACTIONS
<input type="checkbox"/>	CSV	diabetes.csv	Data Asset	Jorge Garza-Ulloa	Oct 08, 2021, 12:40 PM

Click on the button "Add to project"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

13 In the IBM Watson Studio "My Projects/AI and Cognitive Models" section "Add to project"
Note: Click on the button "Modeler flow"

13. In the IBM Watson Studio "My Projects / AI and Cognitive Models" section "Add to project"



Click on the button "Modeler flow"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

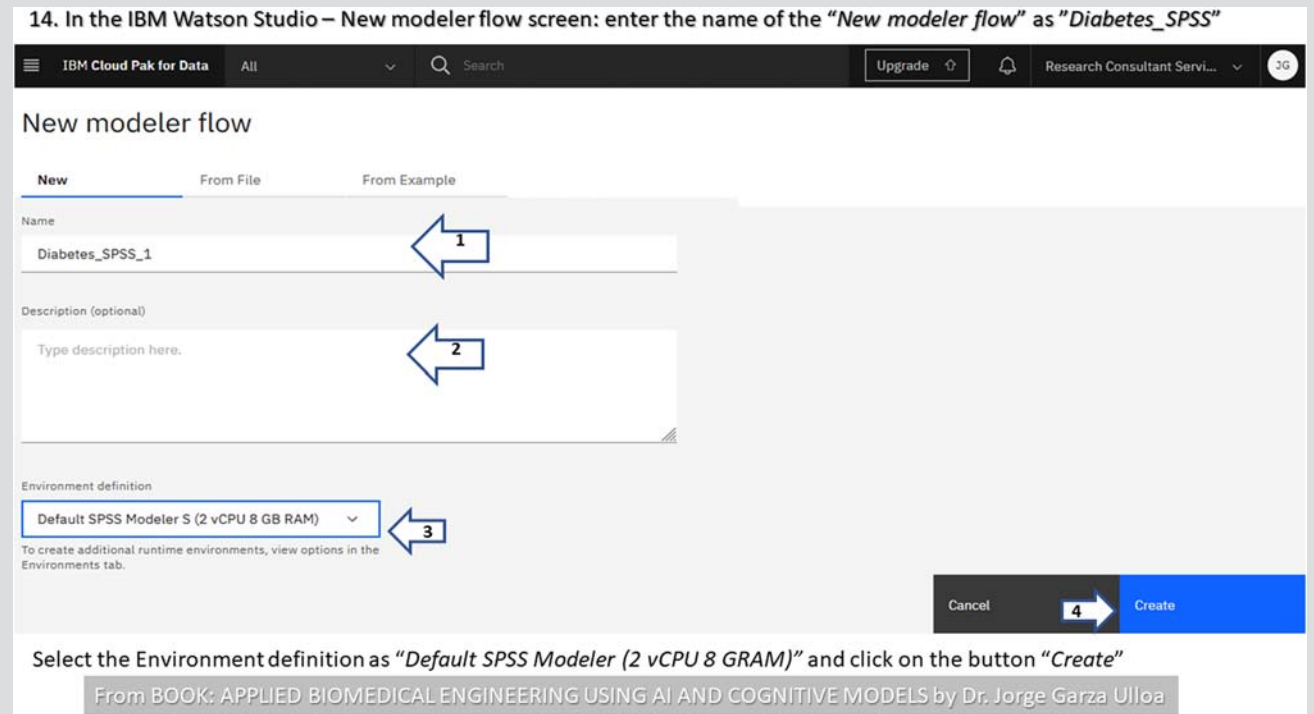
(Continued)

Slide Description

Screen figure

14 In the IBM Watson Studio—New modeler flow screen: enter the name of the “New modeler flow” as “Diabetes_SPSS”
Note: “Select the Environment definition as “Default SPSS Modeler (2 vCPU 8 GRAM)” and click on the button “Create”

14. In the IBM Watson Studio – New modeler flow screen: enter the name of the “New modeler flow” as “Diabetes_SPSS”



Select the Environment definition as “Default SPSS Modeler (2 vCPU 8 GRAM)” and click on the button “Create”

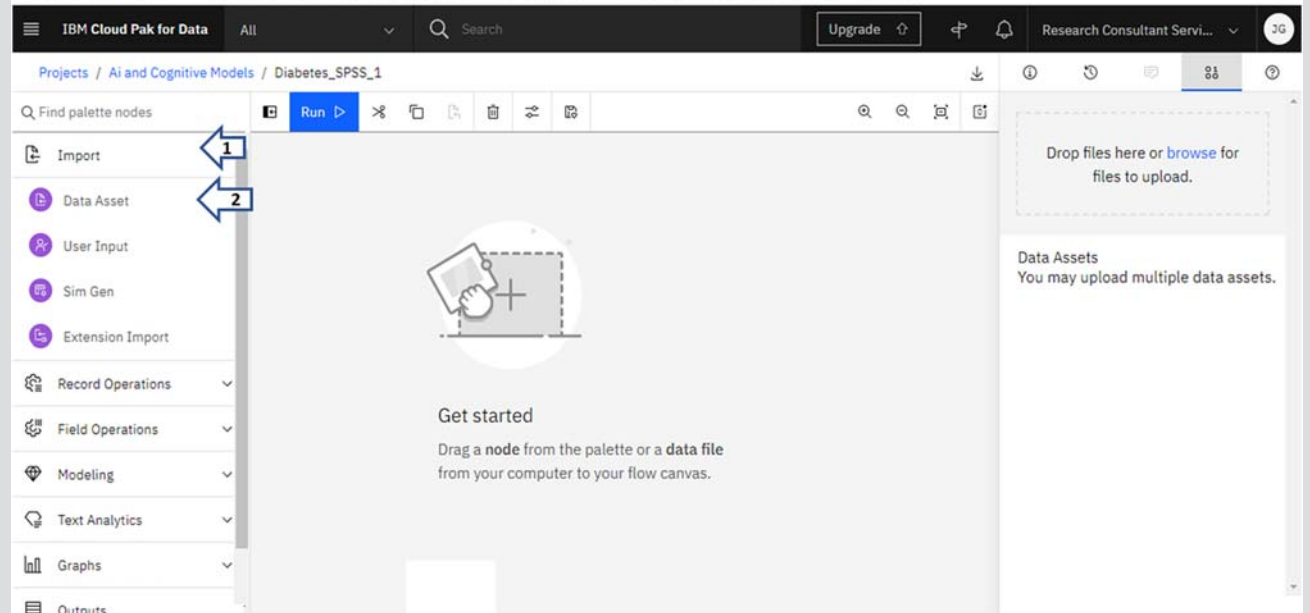
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

15

In the IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen

Note: Select "Import" to expand its menu, click and drag "Data Asset" to create the first node of the model flow

15. In the IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen



Select "Import" to expand its menu, click and drag "Data Asset" to create the first node of the model flow

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

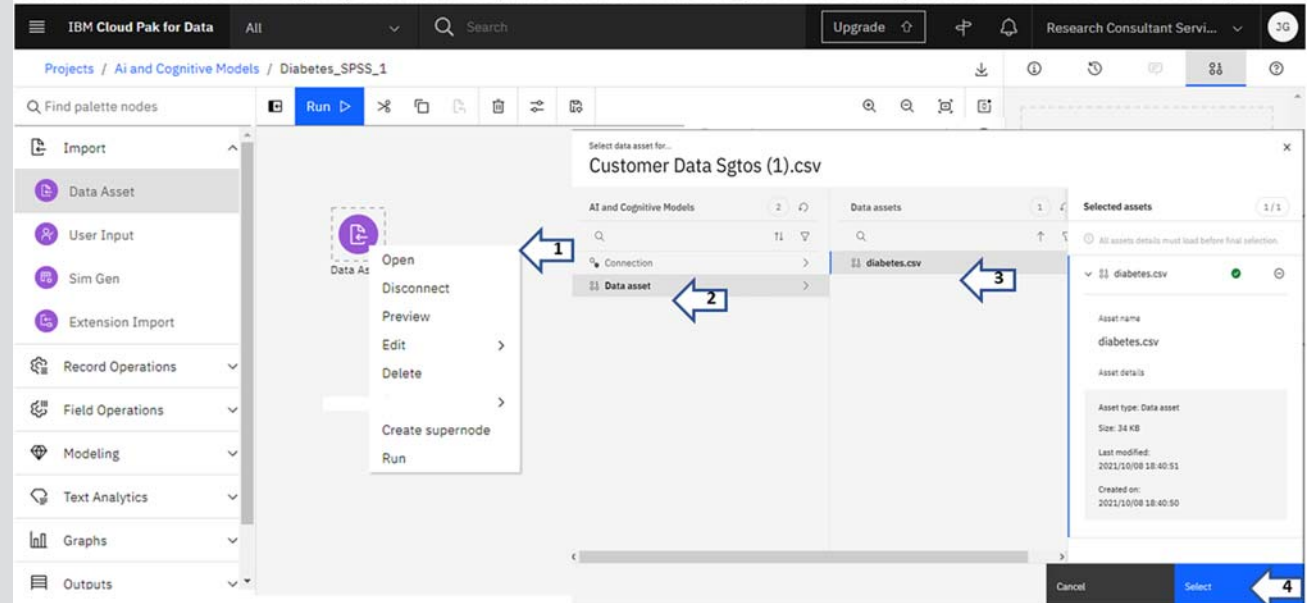
(Continued)

Slide Description

Screen figure

16 In the IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: right click "Data Asset" & select "Open"
Note: Right click on "Data Asset" icon and select "Open" the button to select "Data asset" option for "diabetes.csv" as shown

16. In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: right click "Data Asset" & select "Open"

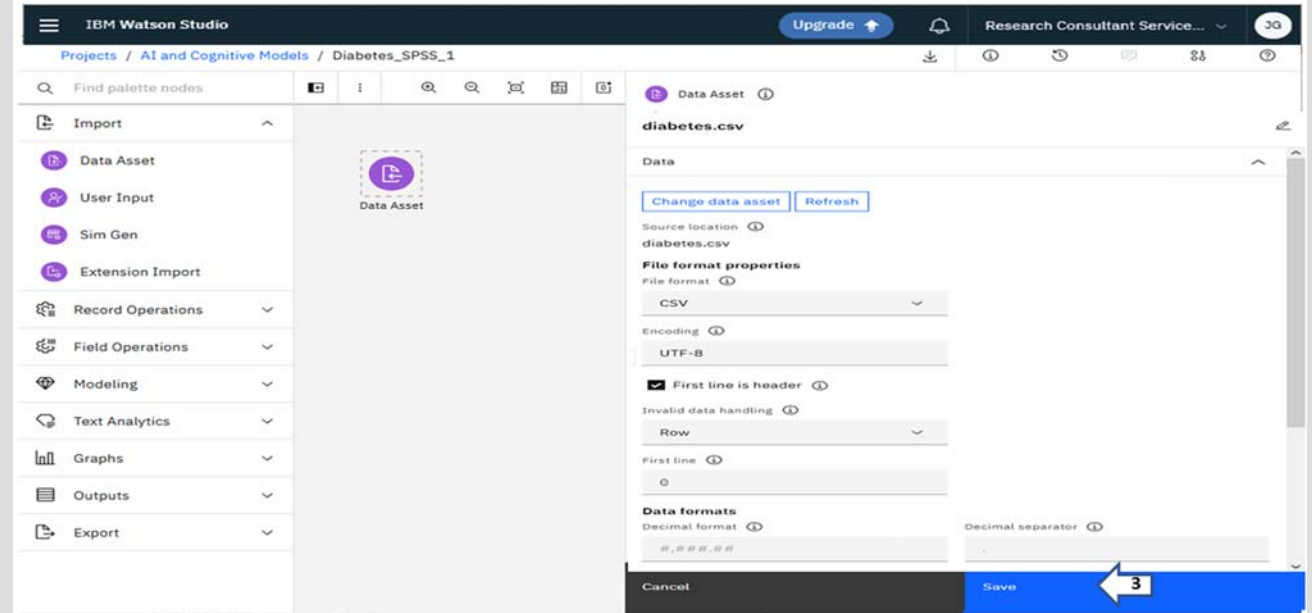


Right click on "Data Asset" icon and select "Open" the button to select "Data asset" option for "diabetes.csv" as shown

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

17 In the IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen, click "Data assets"
Note: Select "diabetes.csv" then select the button "Save"

17. In the IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen, click "Data assets"



Select "diabetes.csv" then select the button "Save"

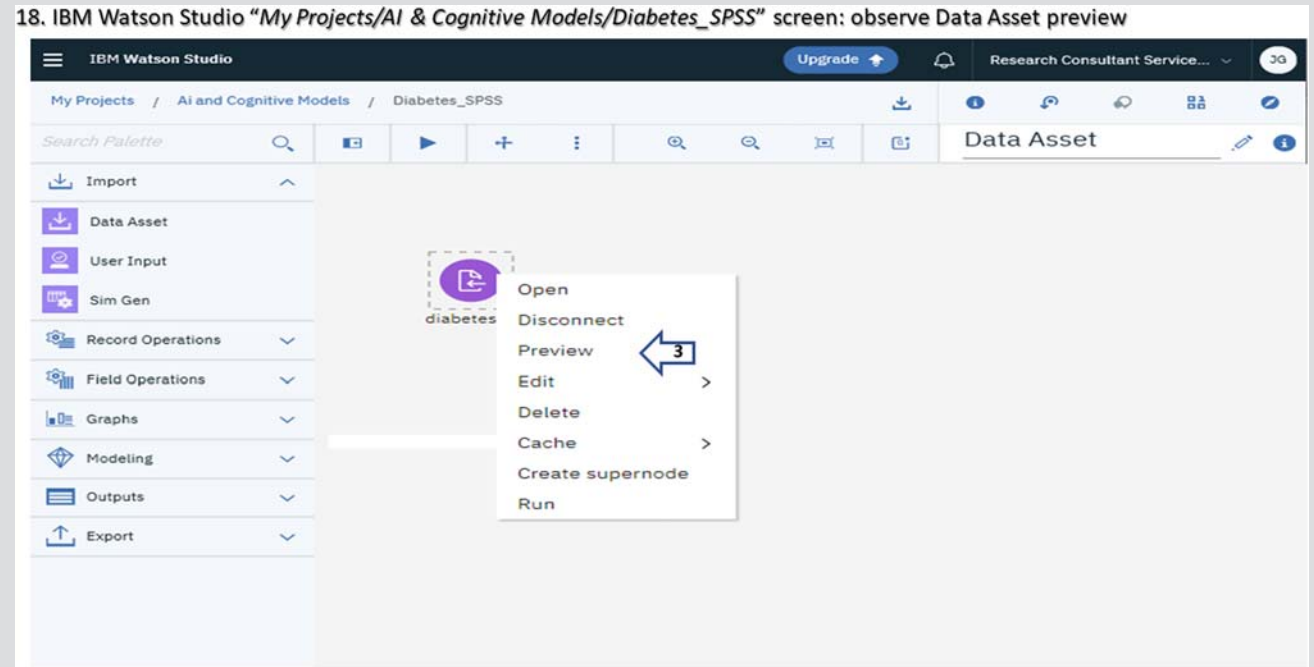
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 18 Description IBM Watson Studio "My Projects/AI & Cognitive Models/Diabetes_SPSS" screen: observe Data Asset preview Note: Make a right click on "Data Asset" and select "Preview"

Screen figure

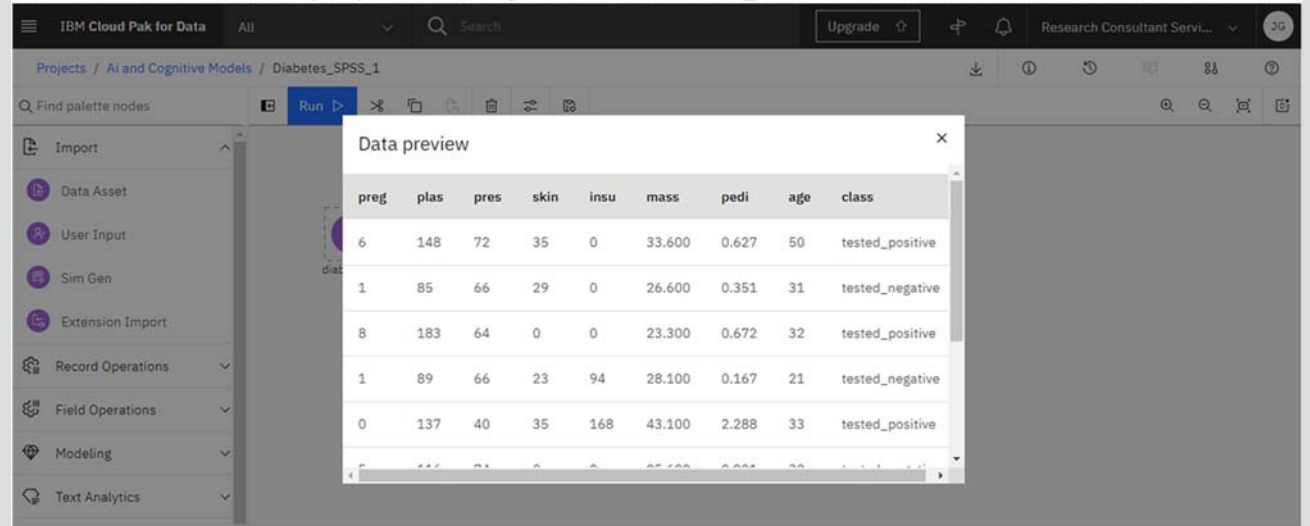


Make a right click on "Data Asset" diabetes icon and select "Preview"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

19 In IBM Watson Studio “My Projects/AI and Cognitive Models/Diabetes_SPSS” screen: after selection of Data Asset > Preview
 Note: Be familiar with the attribute of each field for the “Diabetes.csv”

19. In IBM Watson Studio “My Projects/AI and Cognitive Models/Diabetes_SPSS” screen: after selection of Data Asset >Preview



Be familiar with the attribute of each field for the “Diabetes.csv”

pres	Diastolic blood pressure (mm Hg)
skin	Triceps skin fold thickness (mm)
Insu	2-Hour serum insulin (mu U/ml)
mass	Body mass index (weight in kg/(height in m)^2)
pedi	Diabetes pedigree function
age	Age (years)
class	Class variable ('tested_negative', 'tested_positive')

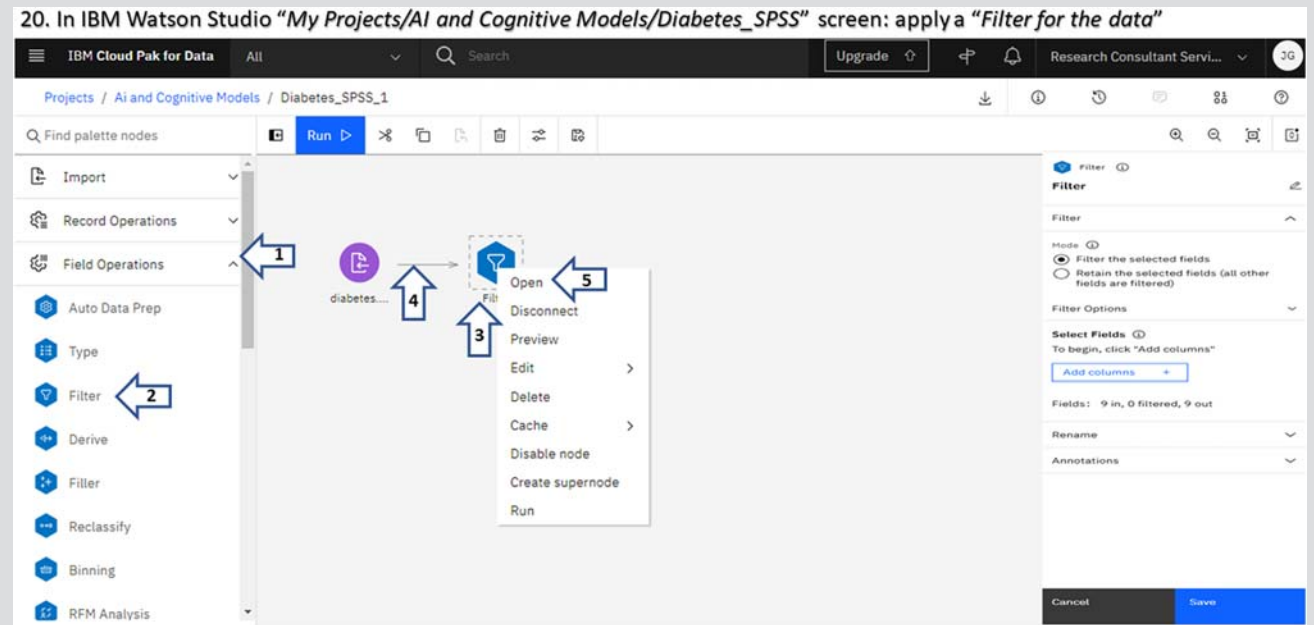
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 20
Description
In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: apply a "Filter for the data"
Note: Extent menu for: "Field Operations," drag "Filter" icon to the Model Flow, joint the nodes, right click "Filter" and select "Open"

Screen figure



Extent menu for: "Field Operations", drag "Filter" icon to the Model Flow, joint the nodes, right click "Filter" and select "Open"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

21 In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: "Filter" node option
Note: Select "Retain the select fields", "Toggle All Fields". Also indicated the 4 fields of the dataset: "press," "mass," "insu," and "class" using "Add Columns," and press the button "Ok"

21. In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: "Filter" node option

The screenshot displays the IBM Watson Studio interface for configuring a 'Filter' node. The left sidebar shows a list of data preparation nodes: Import, Record Operations, Field Operations, Auto Data Prep, Type, Filter, Derive, Filler, Reclassify, Binning, and RFM Analysis. The central workspace shows the 'Filter' node configuration for a dataset named 'diabetes...'. The 'Filter' node has two radio button options: 'Filter the selected fields' (selected) and 'Retain the selected fields (all other fields are filtered)'. Below these are buttons for 'Toggle All Fields', 'Include All Fields', 'Remove All Fields', and 'Remove Duplicates'. The 'Select Fields' section shows a list of fields with checkboxes: 'mass', 'pedi', 'age', and 'class' are checked. The 'Fields' summary indicates '9 in, 0 filtered, 9 out'. The bottom right of the configuration panel has 'Cancel' and 'Save' buttons. A separate 'Select Fields for Filter' panel on the right lists various fields with their data types and checkboxes. The 'OK' button is highlighted with a blue background and a white arrow labeled '5'.

Field	Type	Selected
preg	# integer	<input type="checkbox"/>
plas	# integer	<input type="checkbox"/>
pres	# integer	<input checked="" type="checkbox"/>
skin	# integer	<input type="checkbox"/>
insu	# integer	<input checked="" type="checkbox"/>
mass	# double	<input checked="" type="checkbox"/>
pedi	# double	<input type="checkbox"/>
age	# integer	<input type="checkbox"/>
class	# string	<input checked="" type="checkbox"/>

Select "Retain the select fields", "Toggle All Fields". Also indicated the 4 fields of the data set: "press", "mass", "insu" and "class" using "Add Columns", press the button "Save", and finally "ok"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

22 In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Using a "Field Operations > Type" node
Note: Extent menu: "Field Operations," drag "Type" icon to Model Flow, joint the node with "Filter;" right click "Type" and select "Open"

22. In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS screen": Using a "Field Operations > Type" node

Extent menu: "Field Operations", drag "Type" icon to Model Flow, joint the node with "Filter", right click "Type" and select "Open"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

23 In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Using a "Field Operations> Type" open
Note: With "Filter Type" open specify the role of the field "class" as a Target and press the button "Save"

23. In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Using a "Field Operations> Type" open

The screenshot shows the IBM Watson Studio interface for the project "Diabetes_SPSS_1". The "Type" configuration panel is open, showing the following settings:

- Default Mode: Read metadata, Pass (do not scan)
- Type Operations:
- Find in column Field:
- Table with columns: Field, Measure, Role, Value Mode, Values, Check

Field	Measure	Role	Value Mode	Values	Check
<input type="checkbox"/> # pres	Continuous	Input	Read	↓	None
<input type="checkbox"/> # insu	Continuous	Input	Read	↓	None
<input type="checkbox"/> # mass	Continuous	Input	Read	↓	None
<input type="checkbox"/> # class	Categorical	Target	Read	↓	None

At the bottom of the panel, there are "Cancel" and "Save" buttons. The "Save" button is highlighted with a blue bar and a white box containing the number "2". An arrow labeled "1" points to the "Target" role dropdown for the "class" field.

With "Filter Type" open specify the role of the field "class" as a Target and press the button "Save"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 24 Description In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Using a "Graphs > Distribution" node Note: Extent menu: "Graphs," drag "Distribution" icon to Model Flow, joint the node with "Type," right click "Class" and select "Open," specify "Class" as a name. "Plot Selected fields: Filed (discrete)," press the button "Save" and run the model flow

Screen figure

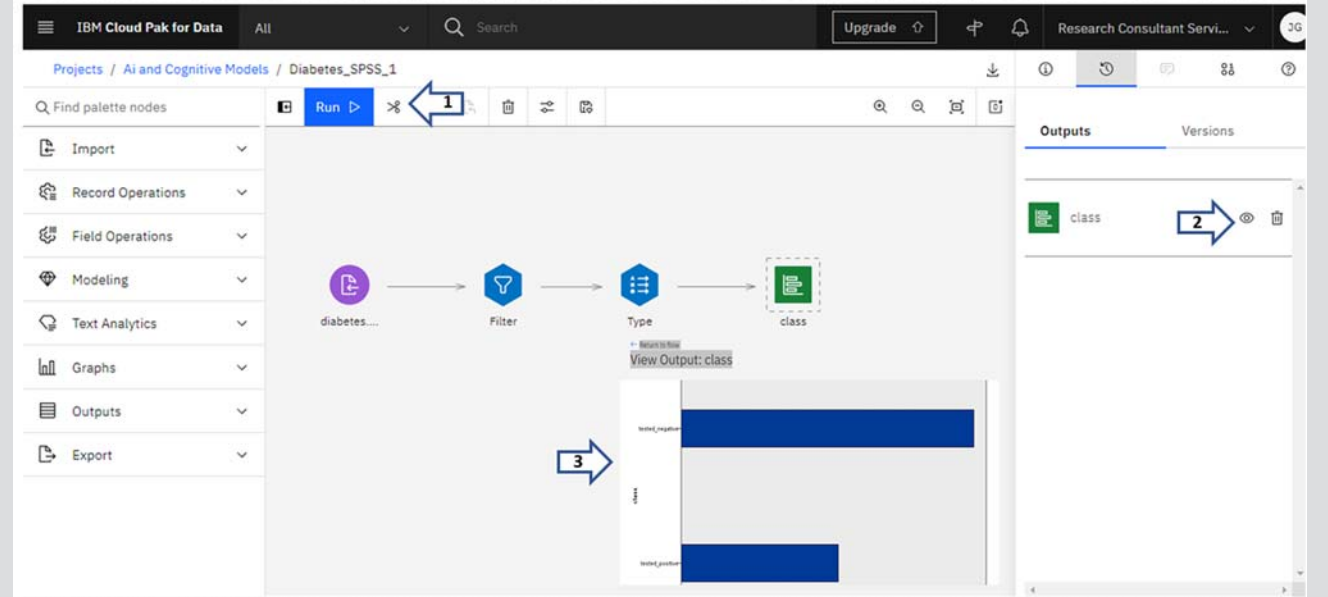
24. In IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Using a "Graphs > Distribution" node

Extent menu: "Graphs", drag "Distribution" icon to Model Flow, joint the node with "Type", right click "Class" and select "Open", specify "Class" as a name, "Plot Selected fields: Filed(discrete)", press the button "Save" and run the model flow

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

25 In IBM Watson Studio “My Projects/AI and Cognitive Models/Diabetes_SPSS” screen: After Run Model flow for “class” distribution chart.
Note: After running the Model Flow an Output class is shown, click on eye symbol and a “Bar graph” is opened in another screen with the distribution of the field “class” the count of “tested_negative = 500” and “tested_positive = 299” for this diabetes dataset

25. IBM Watson Studio “My Projects/AI and Cognitive Models/Diabetes_SPSS” screen: After Run Model flow for “class” distribution



After running the Model Flow an Output class is shown , click on “2” and a “Bar graph” is opened in another screen with the distribution of the field “class” the count of “tested_negative=500” and “tested_positive=299” for this diabetes data set.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

26 IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Run Model flow for "age," "pedi & class graph"
Note: Extent menu: "Graphs," drag "Plot" icon to the Model Flow, joint the node with "diabetes," right click "Plot" and select "Open," Specify "Plot" selecting "3-D graph," assign: "x-field = age," "y-field = pedi," "z-field = class," press the button "Save," and with right click on "Multiplot" node select "Run" the model flow for this node

Screen figure

26. IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Run Model flow for "age, pedi & class" graph

Extent menu: "Graphs", drag "Plot" icon to the Model Flow, joint the node with "diabetes", right click "Plot" and select "Open", Specify "Plot" selecting "3-D graph", "assign: x-field=age", "y-field=pedi", z-field= class", press the button "Save", and with right click on "Multiplot" node select "Run" the model flow for this node

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

27

IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Run Model flow for "age," "pedi & class" graph

Note: After running the Model Flow from the node "Plot" select on the Outputs "age vs pedi vs class," double click it and a "3D graph" is shown that the "Pedi" values are frequently higher for "class = tested_positive" than "class = tested_negative" in this dataset

27. IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Run Model flow for "age, pedi & class" graph

The screenshot displays the IBM Watson Studio interface. On the left, a sidebar contains a 'Find palette nodes' search bar and a list of categories: Import, Record Operations, Field Operations, Modeling, Text Analytics, Graphs, Outputs, and Export. The main workspace shows a model flow with nodes: 'diabetes...', 'Filter', 'Type', and 'class'. A 'Plot' node is highlighted with a blue arrow labeled '1'. Below the flow, an 'Outputs' panel lists: 'All results', 'age v. pedi v. class' (29 seconds ago), and 'Class' (29 seconds ago). A context menu is open over the 'age v. pedi v. class' output, with options: Open, Disconnect, Preview, Edit, Delete, Create supernode, Save branch as a model, and Run. A blue arrow labeled '2' points to the 'Run' option. A blue arrow labeled '3' points to the 'age v. pedi v. class' output entry. On the right, a 'View Output: age v. pedi v. class' window shows a 3D scatter plot with axes labeled 'age', 'pedi', and 'class'. The plot shows a cluster of blue data points. A blue arrow labeled '4' points to the 'View Output' window title.

After running the Model Flow from the node "Plot" select on the Outputs "age vs pedi vs class", click it on open and a "3D graph" is shown that the "Pedi" values are frequently higher for "class=tested_positive" than "class=tested_negative" in this data set. From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

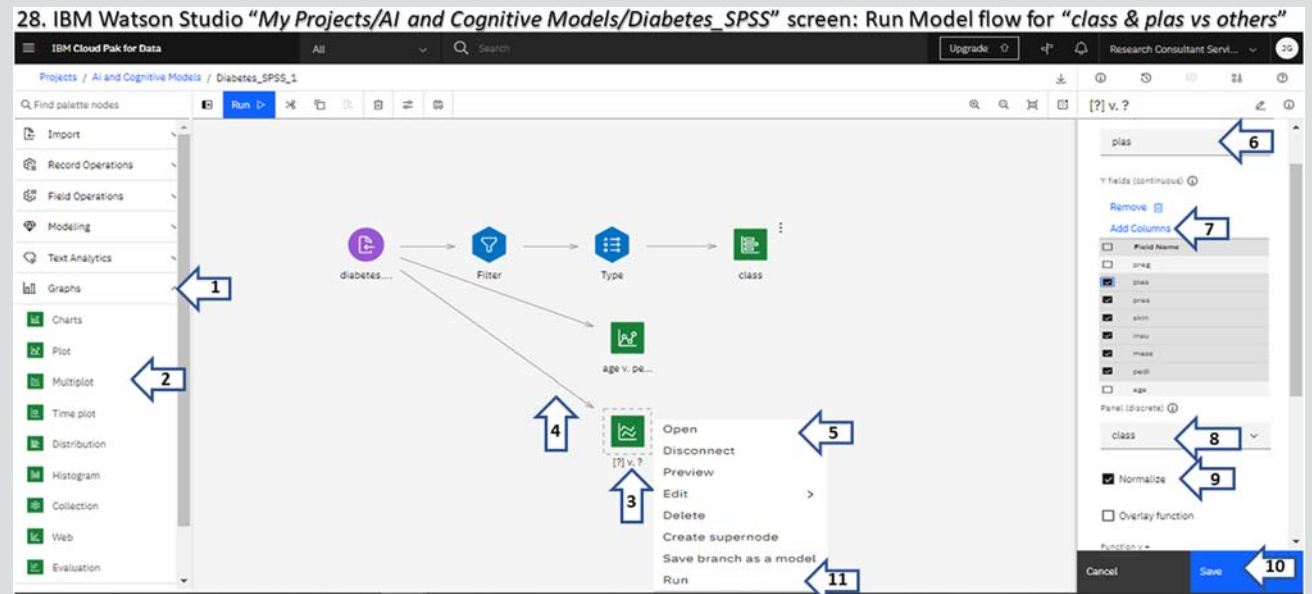
(Continued)

(Continued)

Slide Description

28 IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen: Run Model flow for "class & plas vs others"
Note: Extent menu: "Graphs," drag "Multiplot" icon to the Model Flow, joint the node with "diabetes," right click "Multiplot" node and select "Open," specify "Plot" assigning: "x-field = plas," "y-field = pedi, mass, insu, skin, pres, and preg," activate "panel = normalize," press the button "Save" and with right click on "Multiplot" select "run" the model flow for this node

Screen figure



Extent menu: "Graphs", drag "Multiplot" icon to the Model Flow, joint the node with "diabetes", right click "Multiplot" node and select "Open", specify "Plot assigning x-field=plas", "y-field=pedi, mass, insu, skin, pres, and preg", activate "panel=normalize", press the button "Save" and with right click on "Multiplot" select "run" the model flow for this node.

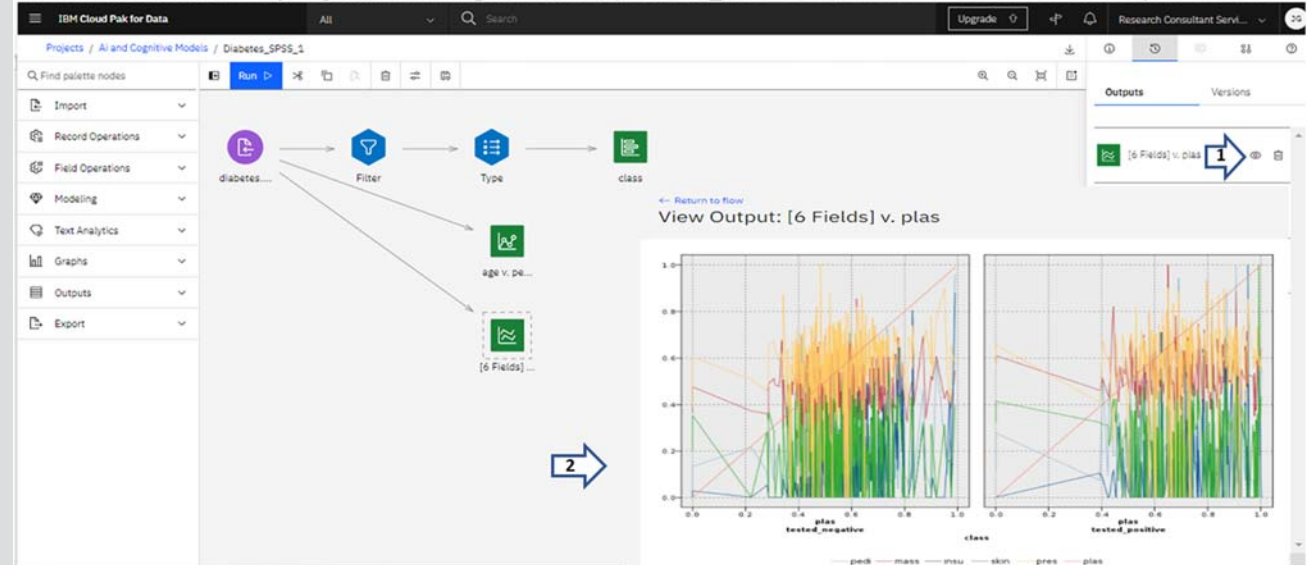
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

29

IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen. Run Model flow for "class & plas vs others"

Note: After running the Model Flow from the node "Multiplot" select on the Output "6 Fields vs plas," double click it and a "Multiplot" in another screen shows that the larger values of "plas" tends to greater likelihood for "testing_positive" than "testing_negative" in diabetes for this particular and small dataset

29. IBM Watson Studio "My Projects/AI and Cognitive Models/Diabetes_SPSS" screen. Run Model flow for "class & plas vs others"



After running the Model Flow from the node "Multiplot" select on the Output "6 Fields vs plas", double click it and a "Multiplot" in another screen shows that the larger values of "plas" tends to greater likelihood for "testing_positive" than "testing_negative" in diabetes for this particular and small data set.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

30 IBM Watson Studio My Projects/AI and Cognitive Models
Note: Select "AI and Cognitive Models" and observe that "Data assets with diabetes.csv," "Modeler flow with Diabetes_SPSS." Log out clicking in the upper right corner of the screen and finally "Log out"

Screen figure

30. IBM Watson Studio "My Projects/AI and Cognitive Models"

The screenshot displays the IBM Watson Studio interface for the project "AI and Cognitive Models". The top navigation bar includes "My Projects / AI and Cognitive Models" (with a callout '1' pointing to the breadcrumb), "Launch IDE", "Add to project", and user information for "Jorge Garza-Ulloa" (with a callout '4' pointing to the user name and a '3G' icon). The main content area is divided into "Data assets" and "Modeler flows". The "Data assets" section shows a table with one asset: "diabetes.csv" (Type: Data Asset, Created by: Jorge Garza-Ulloa, Last modified: 1 Jan 2020, 11:28:07 am). A callout '2' points to the checkbox for this asset. The "Modeler flows" section shows a table with one flow: "Diabetes_SPSS" (Type: SPSS Modeler, Created by: Jorge Garza-Ulloa, Last modified: 1 Jan 2020, 11:32:32 am). A callout '3' points to this flow. A callout '5' points to the "Log out" button in the user profile dropdown menu in the top right corner.

NAME	TYPE	CREATED BY	LAST MODIFIED	ACTIONS
diabetes.csv	Data Asset	Jorge Garza-Ulloa	1 Jan 2020, 11:28:07 am	

NAME	TYPE	CREATED BY	LAST MODIFIED	ACTIONS
Diabetes_SPSS	SPSS Modeler	Jorge Garza-Ulloa	1 Jan 2020, 11:32:32 am	

Select "AI and Cognitive Models" and observe that "Data assets with diabetes.csv", "Modeler flow with Diabetes_SPSS". Log out clicking in the upper right corner of the screen and finally "Log out"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

Using the “IBM Model flow” for a basic analysis for this specific “diabetes.csv” dataset, we observed the following conclusions:

- “Distribution plot” of “class” attribute shown in Fig. 3.16A: there are a total instance of 500 for “tested_negative equal to 65.1%,” and 268 for “tested_positive equal to 34.9%.”
- “3D-plot” of “age vs. class vs. pedi” as shown in Fig. 3.16B: there are frequently higher values of “pedi” with “class = tested_positive” with [min = 0.088, max = 2.42, average = 0.55, std-dev = 0.372] than “class = tested_negative” with [min = 0.078, max = 2.33, average = 0.43, std-dev = 0.299].
- Based on the “normalized multiline plots” of $x = \text{“plas”}$ versus $y = [\text{“pedi,” “mass,” “insu,” “skin,” “pres,” “preg”}]$ separated by “class” as shown in Fig. 3.16C and D, it indicates that larger values of “plas” tends to show a greater likelihood of “testing_positive” for diabetes.

Recommendations

- Update the dataset to include the following diabetes attributes: “urine_test” and “hemoglobin_A1c_test.”
 - “urine_test” may be done in people with diabetes to evaluate severe “hyperglycemia,” that is, severe high blood sugar, by looking for “ketones” in the urine. “Ketones” are a metabolic product produced

when fat is metabolized. “Urine tests” are also done to look for the presence of protein in the urine, which is a sign of kidney damage [19] [https://www.medicinenet.com/urine_tests_for_diabetes/article.htm]

- “hemoglobin_A1c_test” tells our average level of blood sugar over the past 2–3 months. It is also called “HbA1c,” “glycated hemoglobin test,” and “glycohemoglobin.” People who have diabetes need this test regularly to see if their levels are staying within range. For people without diabetes, the normal range for the “hemoglobin A1c” level is between 4% and 5.6%. “Hemoglobin A1c” levels between 5.7% and 6.4% mean you have a higher chance of getting diabetes. Levels of 6.5% or higher mean you have diabetes [20].
- A larger dataset will help to analyze with more precision the diabetes.
- Apply “Machine Learning (ML)” algorithms for classification in a dataset that describes a population that is under a high risk of the onset of diabetes [21].

In this example the IBM Watson Studio was used for a “basic analysis of a diabetes dataset to find relations between their variables applying IBM SPSS Modeler Flow,” the modeling nodes will be explained in Chapter 4, Machine Learning Models Applied to Biomedical Engineering.

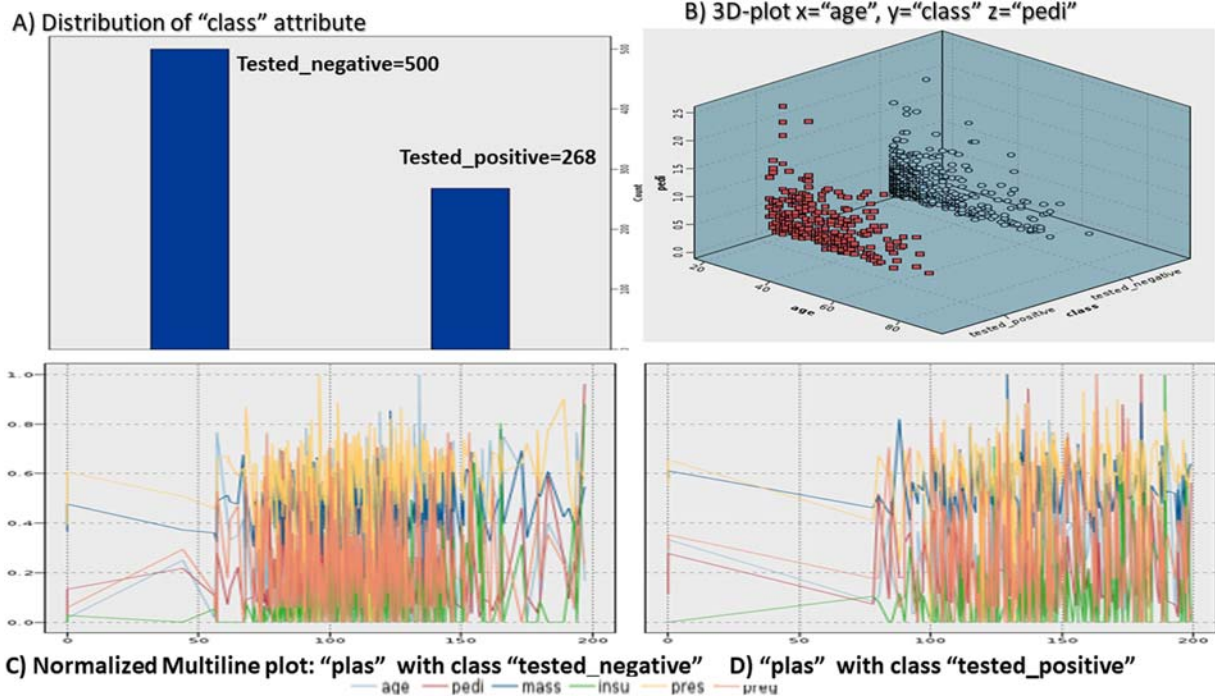


FIGURE 3.16 Plots generated in IBM Watson Studio using IBM SPSS Modeler Flow for a “Basic analysis of a diabetes dataset to find relations between their variables.”

3.5 Examples of applications of evolutionary algorithms with other AI tools in biomedical engineering

The applications of “*evolutionary algorithms*,” such as the “*Genetic Algorithms*” based on “*Darwinian evolution*,” are endless in “*Biomedical Engineering*,” especially in “*Neurology*,” “*Radiology*,” “*Oncology*,” “*Cardiology*,” “*Radiotherapy*,” “*Endocrinology*,” “*Health care management*” and many other fields of medicine [22]. Some examples are:

- “*Neurology*” has many applications of “*Evolutionary Algorithms*.” The most frequently used are:
 - “*Detection of Brain lesion*, such as the ones formed in *Multiple sclerosis (MS)*,” which is a debilitating inflammatory disease in the neural system that forms scars in the brain’s white matter, these scars are known as “*plaques*” [23].
 - “*Detection of seizures in electroencephalogram (EEG)*.” A test records the electrical signals of the brain, and seizure can be detecting by analyzing abnormal brain electrical discharges. “*GA*” is used to find the optimal parameters for and architecture of the “*Artificial Neural networks (ANN)*” [24].
 - “*Detection of mitochondrial dysfunctions*” that play an important role in “*Parkinson’s disease (PD)*.” “*GA*” are used to detect biologically important patterns of mitochondrial mutations in Parkinson’s patients, in addition to the simple comparison of mitochondrial mutations between healthy and disease conditions [25].
 - And many more applications of “*Evolutionary Algorithms* in neurology.”
 - “*Radiology*” is based on use of “*bioinstruments*,” which are machines to scan images from the human body:
 - “*X-rays*” use a form of electromagnetic radiation.
 - “*Computed Tomography (CT)*” uses a computer-processed combination of many X-ray images taken from different angles, to produce cross-sectional images as virtual slices from the body.
 - “*Magnetic Resonance Imaging (MRI)*” uses strong magnetic fields, radio waves, and field gradients to form body image slices.
 - “*Functional magnetic resonance imaging (fMRI)*” uses MRI technology that measures brain activity by detecting changes associated with blood flow.
 - “*Ultrasound imaging*” is a diagnostic imaging technique based on the application of ultrasound as the “*Doppler Ultrasound Analyzer*” devices based on the Doppler effect as a phenomenon, in which an observer perceives a change in the frequency of the sound emitted by the source, when the source, the observer, or both are moving.
 - And many other types of “*imaging bioinstrument machines*.”
- These imaging techniques generate a large amount of data that needs to be analyzed and interpreted by radiologists in a relatively short time. It is necessary to detect the shape and size of objects in these images through edge detection by using “*Evolutionary Algorithms*,” as stated in the following research papers:
- “*Detect macrocalcifications that suggest breast cancer screening primarily using mammography*”: automatic detection of the breast border and nipple position on digital mammograms using a genetic algorithm for an asymmetry approach to detection of microcalcifications [26].
 - “*Mammograms combining wavelet analysis and genetic algorithm*”: segmentation and detection of breast cancer in mammograms combining “*Wavelet Analysis*” and “*Genetic Algorithm*” [27].
 - “*Classification in digital mammograms*”: a “*genetic algorithm*” design for microcalcification detection and classification in digital “*mammograms*” [28].
 - “*Detection of microcalcifications clusters*”: a distributed genetic algorithm for parameters optimization to detect microcalcifications in digital mammograms [29] and a driven genetic algorithm for microcalcification cluster detection [30].
 - “*Detection, analysis y classification of ultrasound breast tumor images*”: combining “*support vector machine*” with “*genetic algorithm*” to classify ultrasound breast tumor images [31].
 - “*Detection of myocardial infarction*”: a “*genetic algorithm* to select variables in logistic regression”: example in the domain of myocardial infarction [32].
 - “*Classification of mass and normal breast tissue*”: image feature selection by a genetic algorithm: application for classification of mass and normal breast tissue [33].
 - “*Detection of solitary lung nodules*” using quality threshold clustering, genetic algorithm, and diversity index [34].
 - And many other applications developed for “*radiology using Evolutionary Algorithms*.”
 - “*Oncology*” uses screening tests for early detection followed by proper treatment that could improve the survival rate of patients. The various methods include:
 - “*Biomolecular information generated via spectroscopy*” that can be analyzed by a “*GA*” partial least square-discriminant analysis system to differentiate between a normal and dysplastic cervix [35].
 - “*Detect massive gene expression used for molecular diagnostics and prognosis using DNA microarrays*.” DNA microarrays generate a large set of data that need to be analyzed to detect the key predictive genes. “*GA*” has the capability to search and find optimal solutions among large and

- complex solutions through many interactions, allowing the detection of many cancer cells [36].
- “*Find relationships between several elements and cancer*”: this can be achieved using a combination of “GA,” “Regression Analysis,” and “Least Square Support Vector Machine (LSSVM) model” to find patterns in training data to predict the mortality of cancer cervical and trace elements [37]. Modeling the relationship between cervical cancer mortality and trace elements is based on genetic algorithm-partial least squares and support vector machines [38].
 - And many other techniques specially developed for “Oncology applying “Evolutionary Algorithms.”
 - “Cardiology” uses “Evolutionary Algorithms” in many fields of cardiovascular medicine, such as:
 - “Detection and analysis of atherosclerotic plaques,” which are the indicators most myocardial infarctions and strokes. Frequently the plaque mechanical properties, such as elasticity, allow to locate the unstable plaques [39].
 - “Model the presence of myocardial infarction in patients with chest pain”: a “Genetic Algorithm” is used to select variables in “Logistic Regression”: an example in the domain of myocardial infarction [32].
 - “Interpretation of the electrocardiogram (ECG) based on the detection of QRS complex to diagnose “Cardiac arrhythmias,” where the “QRS complex is a specific sequence of deflections” seen on the print-out of an “ECG,” representing the depolarization of the right and left ventricles of the heart [40].
 - And many other applications specially design for “Cardiology using Evolutionary Algorithms.”
 - “Radiotherapy” is a treatment where radiation is used to kill cancer cells. These treatments use “bioinstruments” to apply “Intensity modulated radiotherapy (IMRT)” to transfer accurate doses of radiation to a target tumor in the brain, prostate, etc. They damage cancer cells and stop them from growing or spreading in the body. Examples of “Evolutionary Algorithms” are:
 - “Planning IMRT sessions,” which typically involves the selection of 5–10 angles for “wavelet” projection and determining the radiation dose. “GA” could improve the selection of “gantry angles” in a reasonable time frame, where the “gantry of a computed tomography scanner (CT)” is a ring or cylinder, into which a patient is placed. The “X-ray tube” and “X-ray detector” spin rapidly in the gantry, as the patient is moved in and out of the gantry [41].
 - “Calculation and planning of irradiation planning for other types of cancer including pancreatic [42], rhabdomyosarcoma, and brain tumors [43].
 - And many other algorithms based on “Evolutionary Algorithms for precise irradiation planning for different human body organs.”
 - “Endocrinology” is a branch of biology and medicine dealing with the endocrine system, its diseases, and its specific secretions known as “hormones,” that need many special algorithms to:
 - “Detect hypoglycemia,” which is the most common complication of “insulin therapy” in patients with “type 1 diabetes mellitus.” “Hypoglycemia” can induce alterations in the patterns of “Electroencephalograms (EEG)” signals that can be detected by the combination of different “AI tools” as: “Artificial Neural Networks (ANN)” for training, “Genetic Algorithms (GA)” for search and “Levenberg-Marquardt (LM)” training techniques used to solve nonlinear least squares problem [44].
 - “Detect noninvasive episodes of nocturnal hypoglycemia” using “heart rate and corrected QT interval” that represents the time taken for ventricular depolarization and repolarization applying different “AI tools,” such as “GA” based “Multiple Regression” with “Fuzzy Inference System” [45].
 - And many other “AI algorithms for Endocrinology applications.”
 - “Health care management” as the administration, management, or oversight of healthcare systems, public health systems, hospitals, entire hospital networks, or other medical facilities. “AI tools” based on “Evolutionary Algorithms” can be applied for many applications of hospital management to improve patient servicing, satisfaction, and cost-effectiveness ratios as well as the efficient scheduling of patient’s admission and follow-up, such as:
 - “Improve the patient admissions and scheduling in different hospital areas” using a mathematical model developed and optimized applying a “GA” [46], and clinical pathways scheduling using hybrid genetic algorithm [47].
 - “Optimization of clinical laboratories”: “GAs” have been applied to improve staff rotation scheduling in a clinical laboratory, for planning the rotation of staff effectively, ensuring the maintenance of techniques and skills, saving time and the cost necessary for the scheduling process.
 - And many other algorithms based on “Evolutionary Algorithms” for health care management.”

There are many other *Evolutionary Algorithms* that can be applied in other biomedical engineering fields, such as: “Obstetrics and gynecology,” “Pediatrics,” “Surgery,” “Pulmonology,” “Infectious diseases,” “Rehabilitation medicine,” “Orthopedics,” “Pharmacotherapy,” etc. The understanding of the methods explained in this chapter will facilitate the optimization of many AI applications developed for BME.

References

- [1] Available from: <https://www.healthline.com/health/blood-cell-disorders#plasma-cell-disorders> (accessed 12.11.19).
- [2] Available from: https://www.mayoclinic.org/diseases-conditions/brain-tumor/symptoms-causes/syc-20350084?mc_id=bing&campaign=329764198&geo=75576&kw=is%20it%20%2BTumor%20of%20the%20%2BBrain&query=brain%20tumor%20detection%20using%20matlab&ad=80126970698623&network=Search&sitetarget=o&adgroup=1282030750377700&extension=&target=&matchtype=b&device=c&account=B013932Y&invsrc=neuro&placement=enterprise&msclkid=6740ebbb9bab1bc0e5e9b671541f2ab4 (accessed 12.11.19).
- [3] T. Gangavarapu, N. Patil, A novel filter–wrapper hybrid greedy ensemble approach optimized using the genetic algorithm to reduce the dimensionality of high-dimensional biomedical datasets, *Appl. Soft Comput.* 81 (2019) 105538. Available from: <https://doi.org/10.1016/j.asoc.2019.105538>. <http://www.sciencedirect.com/science/article/pii/S156849461930314X>. ISSN 1568-4946.
- [4] M. Lahanas, D. Baltas, N. Zamboglou, A hybrid evolutionary multiobjective algorithm for anatomy based dose optimization algorithm in high-dose-rate brachytherapy, *Phys. Med. Biol.* 48 (3) (2003) 399–415.
- [5] O.E. Kundakcioglu, P.M. Pardalos, Optimization in biomedical research, *Fields Inst. Commun.* 55 (2009).
- [6] C.-H. Lee, M. Schmidt, A. Murtha, A. Bistriz, J. Sander, R. Greiner, Segmenting brain tumors with conditional random fields and support vector machines, *CVBIA* (2005) 469–478.
- [7] A. Genkin, C.A. Kulikowski, I. Muchnik, Set covering submodular maximization: An optimal algorithm for data mining in bioinformatics and medical informatics, *J. Intell. Fuzzy Syst.* 12 (2002) 5–17.
- [8] P. Sharma, K. Choudhary, K. Gupta, R. Chawla, D. Gupta, A. Sharma, Artificial plant optimization algorithm to detect heart rate & presence of heart disease using machine learning, *Artif. Intell. Med.* 102 (2020) 101752. Available from: <https://doi.org/10.1016/j.artmed.2019.101752>. <http://www.sciencedirect.com/science/article/pii/S0933365719304312>. ISSN 0933-3657.
- [9] Y. Zhang, S. Kwong, S. Wang, Machine learning based video coding optimizations: a survey, *Inf. Sci.* 506 (2020) 395–423. Available from: <https://doi.org/10.1016/j.ins.2019.07.096> ISSN 0020-0255. Available from: <http://www.sciencedirect.com/science/article/pii/S0020025519307145>.
- [10] J. Wu, X.-Y. Chen, H. Zhang, L.-D. Xiong, H. Lei, S.-H. Deng, Hyperparameter optimization for machine learning models based on Bayesian optimization, *J. Electron. Sci. Technol.* 17 (1) (2019) 26–40. Available from: <https://doi.org/10.11989/JEST.1674-862X.80904120>. <http://www.sciencedirect.com/science/article/pii/S1674862X19300047>. ISSN 1674-862X.
- [11] D. Chitradevi, S. Prabha, Analysis of brain sub regions using optimization techniques and deep learning method in Alzheimer disease, *Appl. Soft Comput.* (2019) 105857. Available from: <https://doi.org/10.1016/j.asoc.2019.105857>. <http://www.sciencedirect.com/science/article/pii/S1568494619306386>. ISSN 1568-4946.
- [12] S. Vellappally, A.A. Al Kheraif, S. Anil, A.A. Wahba, IoT medical tooth mounted sensor for monitoring teeth and food level using bacterial optimization along with adaptive deep learning neural network, *Measurement* 135 (2019) 672–677. Available from: <https://doi.org/10.1016/j.measurement.2018.11.078>. <http://www.sciencedirect.com/science/article/pii/S0263224118311333>. ISSN 0263-2241.
- [13] T.Y. Tan, L. Zhang, C.P. Lim, Intelligent skin cancer diagnosis using improved particle swarm optimization and deep learning models, *Appl. Soft Comput.* 84 (2019) 105725. Available from: <https://doi.org/10.1016/j.asoc.2019.105725>. <http://www.sciencedirect.com/science/article/pii/S156849461930506X>. ISSN 1568-4946.
- [14] F.B. Ozsoydan, Artificial search agents with cognitive intelligence for binary optimization problems, *Comput. Ind. Eng.* 136 (2019) 18–30. Available from: <https://doi.org/10.1016/j.cie.2019.07.007>. <http://www.sciencedirect.com/science/article/pii/S0360835219303985>. ISSN 0360-8352.
- [15] M.R. Salmanpour, M. Shamsaei, A. Saberi, S. Setayeshi, I.S. Klyuzhin, V. Sossi, et al., Optimized machine learning methods for prediction of cognitive outcome in Parkinson’s disease, *Comput. Biol. Med.* 111 (2019) 103347. Available from: <https://doi.org/10.1016/j.compbiomed.2019.103347>. <http://www.sciencedirect.com/science/article/pii/S0010482519302161>. ISSN 0010-4825.
- [16] A. Kitsantas, A.L. Baylor, S.E. Hiller, Intelligent technologies to optimize performance: Augmenting cognitive capacity and supporting self-regulation of critical thinking skills in decision-making, *Cognit. Syst. Res.* 58 (2019) 387–397. Available from: <https://doi.org/10.1016/j.cogsys.2019.09.003>. <http://www.sciencedirect.com/science/article/pii/S1389041719304693>. ISSN 1389-0417.
- [17] C. Darwin, *On the Origin of Species, or the Preservation of Favoured Races in the Struggle for Life*, John Murray, London, 1859.
- [18] D. Câmara, 1 - Evolution and evolutionary algorithms, in: D. Câmara (Ed.), *Bio-inspired Networking*, Elsevier, 2015, pp. 1–30. Available from: <https://doi.org/10.1016/B978-1-78548-021-8.50001-6>. <http://www.sciencedirect.com/science/article/pii/B9781785480218500016>. ISBN 9781785480218.
- [19] Melissa Conrad Stöppler, M.D., Jerry R. Balentine, DO, FACEP, Diabetes Urine Tests, 11/05/2020. Available from: https://www.medicinenet.com/urine_tests_for_diabetes/article.htm, accessed 10-09-2021.
- [20] Available from: <https://www.webmd.com/diabetes/guide/glycated-hemoglobin-test-hba1c?lastselectedguid=%7b5FE84E90-BC77-4056-A91C-9531713CA348%7d>.
- [21] Available from: <https://machinelearningmastery.com/case-study-predicting-the-onset-of-diabetes-within-five-years-part-1-of-3/>.
- [22] A. Ghaehri, S. Shoar, M. Naderan, S.S. Hoseini, The applications of genetic algorithms in medicine, *Oman Med. J.* 30 (6) (2015) 406–416. Available from: <https://doi.org/10.5001/omj.2015.82>.
- [23] J.L. Jaremko, P. Poncet, J. Ronsky, J. Harder, J. Dansereau, H. Labelle, et al., Genetic algorithm-neural network estimation of cobb angle from torso asymmetry in scoliosis, *J. Biomech. Eng.* 124 (5) (2002) 496–503.
- [24] S. Koçer, M.R. Canal, Classifying epilepsy diseases using artificial neural networks and genetic algorithm, *J. Med. Syst.* 35 (4) (2011).
- [25] R. Smigrodzki, B. Goertzel, C. Pennachin, L. Coelho, F. Prosdoci, W.D. Parker Jr., Genetic algorithm for analysis of mutations in Parkinson’s disease, *Artif. Intell. Med.* 35 (3) (2005) 227–241.

- [26] M. Karman, K. Thangavel, Automatic detection of the breast border and nipple position on digital mammograms using genetic algorithm for asymmetry approach to detection of microcalcifications, *Comput. Meth. Prog. Biomed.* 87 (1) (2007) 12–20.
- [27] D.C. Pereira, R.P. Ramos, M.Z. do Nascimento, Segmentation and detection for breast cancer in mammograms combining wavelet analysis and genetic algorithm, *Comput. Meth. Prog. Biomed.* 114 (1) (2014) 88–101.
- [28] J. Jiang, B. Yao, A.M. Wason, A genetic algorithm design for microcalcification detection and classification in digital mammograms, *Comput. Med. Imag. Graph.* 31 (1) (2007) 49–61.
- [29] A. Bevilacqua, R. Campanini, N. Lanconelli, A distributed genetic algorithm for parameters optimization to detect micro calcifications in digital mammograms, *Appl. Evolut. Comput. Proc.* 2037 (2001) 278–287.
- [30] B. Yao, J.M. Jiang, Y.H. Peng, A CBR driven genetic algorithm for microcalcification cluster detection, *Eng. Knowl. Age Semant. Web. Proc.* 3257 (2004) 494–496.
- [31] W.J. Wu, S.W. Lin, W.K. Moon, Combining support vector machine with genetic algorithm to classify ultrasound breast tumor images, *Comput. Med. Imag. Graph.* 36 (8) (2012) 627–633.
- [32] S. Vinterbo, L. Ohno-Machado, A genetic algorithm to select variables in logistic regression: example in the domain of myocardial infarction, *Proc. AMIA Symp.* (1999) 984–988.
- [33] B. Sahiner, H.P. Chan, D. Wei, N. Petrick, M.A. Helvie, D.D. Adler, et al., Image feature selection by a genetic algorithm: application to classification of mass and normal breast tissue, *Med. Phys.* 23 (10) (1996) 1671–1684.
- [34] A.O. de Carvalho Filho, W.B. de Sampaio, A.C. Silva, A.C. de Paiva, R.A. Nunes, M. Gattass, Automatic detection of solitary lung nodules using quality threshold clustering, genetic algorithm and diversity index, *Artif. Intell. Med.* 60 (3) (2014) 165–177.
- [35] S. Duraipandian, W. Zheng, J. Ng, J.J. Low, A. Ilancheran, Z. Huang, In vivo diagnosis of cervical precancer using Raman spectroscopy and genetic algorithm techniques, *Analyst* 136 (20) (2011) 4328–4336.
- [36] M. Dolled-Filhart, L. Rydén, M. Cregger, K. Jirström, M. Harigopal, R.L. Camp, et al., Classification of breast cancer using genetic algorithms and tissue microarrays, *Clin. Cancer Res.* 12 (21) (2006) 6459–6468.
- [37] C.H. Ooi, P. Tan, Genetic algorithms applied to multiclass prediction for the analysis of gene expression data, *Bioinformatics* 19 (1) (2003) 37–44.
- [38] C. Tan, H. Chen, T. Wu, C. Xia, Modeling the relationship between cervical cancer mortality and trace elements based on genetic algorithm-partial least squares and support vector machines, *Biol. Trace Elem. Res.* 140 (1) (2011) 24–34.
- [39] A.S. Khalil, B.E. Bouma, M.R. Kaazempur Mofrad, A combined FEM/genetic algorithm for vascular soft tissue elasticity estimation, *Cardiovasc. Eng.* 6 (3) (2006) 93–102.
- [40] C. Tu, Y. Zeng, X. Yang, A new approach to detect QRS complexes based on a histogram and genetic algorithm, *J. Med. Eng. Technol.* 29 (4) (2005) 176–180.
- [41] D.P. Nazareth, S. Brunner, M.D. Jones, H.K. Malhotra, M. Bakhtiari, Optimization of beam angles for intensity modulated radiation therapy treatment planning using genetic algorithm on a distributed computing platform, *J. Med. Phys.* 34 (3) (2009) 129–132.
- [42] G.A. Ezzell, L. Gaspar, Application of a genetic algorithm to optimizing radiation therapy treatment plans for pancreatic carcinoma, *Med. Dosim.* 25 (2) (2000) 93–97.
- [43] X. Wu, Y. Zhu, A mixed-encoding genetic algorithm with beam constraint for conformal radiotherapy treatment planning, *Med. Phys.* 27 (11) (2000) 2508–2516.
- [44] L.B. Nguyen, A.V. Nguyen, S.H. Ling, H.T. Nguyen, Combining genetic algorithm and Levenberg-Marquardt algorithm in training neural network for hypoglycemia detection using EEG signals, *Conf. Proc. IEEE Eng. Med. Biol. Soc.* 2013 (2013) 5386–5389.
- [45] S.S. Ling, H.T. Nguyen, Genetic-algorithm-based multiple regression with fuzzy inference system for detection of nocturnal hypoglycemic episodes, *IEEE Trans. Inf. Technol. Biomed.* 15 (2) (2011) 308–315.
- [46] L.M. Zhang, H.Y. Chang, R.T. Xu, The patient admission scheduling of an ophthalmic hospital using genetic algorithm, *Adv. Mat. Res.* 756 (2013) 1423–1432.
- [47] G. Du, Z. Jiang, Y. Yao, X. Diao, Clinical pathways scheduling using hybrid genetic algorithm, *J. Med. Syst.* 37 (3) (2013) 9945.

This page intentionally left blank

Machine Learning Models Applied to Biomedical Engineering

4.1 Introduction

As explained in Section 1.4, “Machine Learning (ML)” is a subset of “AI” that evolved from the study of “pattern recognition and computational learning theory.” “ML” has seven specific steps to follow to achieve its goal of obtaining a valid model for prediction, these steps were explained in Section 1.3.8 and summarized in Fig. 1.4; these are “data collection,” “data preparation and exploration,” “feature engineering,” “model selection,” “model training,” “model evaluation,” and “model prediction and deployment”:

Step 1) “Data collection” is made under two terms: “data schema” and “semantic types.”*

Step 2) “Data preparation and exploration” is the cleaning of data collected, and data exploration is the tools to avoid the problems of “AI bias.”*

Step 3) “Feature engineering” consists of the feature selection and includes the deletion of irrelevant attributes or even the creation of new features, “apply dimensionality using Principal Component Analysis (PCA)” and “finally split the dataset into training and evaluation sets.”*

Step 4) “Model selection” refers to the selection of the best algorithm and the platform, which could be “open source language,” “high-performance language,” “special AI cloud applications,” or others.*

Step 5) “Model training” is running the algorithm to obtain a process model based on iteration as a training step using the actual dataset, to train the model for performing various actions, model evaluation, and model prediction and deployment.*

Step 6) “Model evaluation and tuning” are the use of metrics or a combination of them to measure the objective performance of the model and to select the best algorithm for that specific purpose.*

Step 7) “Model prediction and deployment” is where the “ML model” obtained is used to predict the outcome of what we need, and the “deployment” refers to*

the creation of applications of a model for predictions using a new data where it is needed.

“Machine Learning (ML)” is a subset of “AI” that evolved from the study of “pattern recognition and computational learning theory.” “ML” has seven specific steps to follow to achieve its goal of obtaining a valid model for prediction, these steps were explained in Section 1.3.8 and summarized in Fig. 1.4.

4.2 Choosing the best ML model

Each “ML model” has its strengths and weakness for each specific scenario; “there are no rules of thumb or easy steps to follow.” The basic questions for narrowing the selection of the “ML model” to use are the “data to analyze,” “task(s) to accomplish,” “software type,” “hardware needed,” “deployment needed,” and “type of validation.” Where:

- “Data to analyze”:
 - Type of data: “discrete values” or “continuous.”
 - Data complexity: “straightforward or complex.” Allows try to detect and ignore irrelevant attributes.
 - Size of the datasets: “small, medium, or large.”
- “Task to accomplish”:
 - Describe “exactly what we want to accomplish.”
 - Visualize: “have a clear visualization of the process needed before proceed.”
 - Detail needed: “define the amount of detail needed in the results.”
- “Software type”:
 - Select computer language (free AI and general-purpose programming language), such as open source language, for example, “cURL” (command lines or scripts to transfer data), “Java,” “JavaScript,” “Python,” “Scala,” “R language” (free AI data-statistical analysis programming language), or others.

- *High-performance language* such as “MATLAB®,” “Lisp,” “Prolog,” others.
- *Special AI cloud computing services* such as “IBM Cloud Watson Machine,” “Amazon Web Services (AWS),” “Microsoft Azure,” and others.
- “*Hardware needed*”:
 - *Processing power*: limited, normal, excessive.
 - *Storage needed*: limited, normal, excessive.
- “*Deployment needed*”:
 - *Locally, private, public, hybrid, community cloud, or other.*
 - *Need real time?*
 - *Monouser or multiuser?*
- “*Type of validation*”: “*Validation gives a numeric estimation of the difference between the actual data in the dataset and the estimated result,*” but it is important to mention that this is performance is obtained only with the data used to train it. Then, this is not a guarantee that the model is either “*overfitting*” or “*underfitting*.” “*Overfitting*” is when the “*ML model*” starts learning from the noise and inaccurate data that probably will lead to obtaining a wrong prediction, and “*underfitting*” is when the “*ML model*” cannot capture the underlying trend of the data. “*Cross-validation*” is a technique to evaluate “*ML algorithm’s performance*” based on a partition of dataset in a subset for training and the other subset for testing. There are two general methods for classification of “*cross-validation*”: “*non-exhaustive*” and “*exhaustive*”.
 - “*Nonexhaustive,*” where the data is trained only for some combinations from the dataset. The most common methods are the following:
 - “*Holdout method*”: the easiest “*cross-validation*” method that splits the entire dataset into two sets, a “*training set*” and a “*test or validation set.*” The problem is that any data could be in the “*training set*” or in the “*validation set,*” and have a high variance. Note: Variance on statistics measures variability from the average or mean
 - “*K-Fold cross-validation*”: the original training set is divided into “*k subsets.*” In each fold, one of the “*k subsets*” is taken as the “*validation set,*” and the “*remaining k – 1 subsets* are used as the *training set.*” The “*error estimations from all the folds*” are taken and “*averaged to give us the final error estimation of the model,*” based in that all the “*k sets for validation,*” each datapoint appears in the validation set exactly once. And each point of data appears in the training set exactly “*k – 1*” times, then the accuracy of the “*ML model*” is improved.
 - “*Stratified K-fold cross-validation*”: the difference is that in each set a verification is made

that in each category there are equal or close results and/or the “*means of all outcomes are comparable.*”

- “*Exhaustive*”: data is trained for all possible combinations. The most common methods are:
 - “*Leave-P-out cross-validation*”: the “*n*” data points are divided into “*n – p*” that are taken in one iteration and the remaining “*p*” are used for validation, where the iteration checks all possible combination of “*p.*”
 - “*Leave-one-out cross-validation*”: very similar to the “*Leave-P-out cross-validation*” but “*p*” is always “*1,*” with the advantage that this method is less “*exhaustive.*”

The “*ML Model*” to select to find the solution depends on the type of ML problem; generally this can be “*unsupervised learning,*” “*supervised learning,*” “*reinforcement learning,*” “*survival models,*” “*association rules,*” and others.

4.2.1 Unsupervised learning

“*Unsupervised learning*” is used when the data do not include the result for each case, meaning that the ground truths are unknown. “*Unsupervised Learning Models*” have the ability to find “*patterns in a stream of inputs*” known as “*clusters,*” because they are in groups and interpreted based only on input data without “*labels*” responses. “*Clustering* is used for exploratory data analysis to find hidden patterns or grouping of the data.” In “*biomedical engineering*” it is used for many application as “*gene sequence analysis,*” “*motion detection,*” “*Natural Language Processing (NLP),*” and as “*computational biology*” for tumor detection, drug discovery, etc. The “*ML models*” frequently used as “*Unsupervised Learning*” are indicated in Fig. 4.1, they are: “*k-means*,*” “*k-Medoids,*” “*Hierarchical,*” “*Gaussian Mixture,*” “*Artificial Neural Networks,*” “*Hidden Markov,*” “*Self-organizing map,*” and others.

Note*: See “*k-means*” applied on: “*Research 4.1 Tutorial example IBM Watson SPSS Modeler Flow for ML Model for Diabetes.*”

4.2.2 Supervised learning

“*Supervised Learning*” is used when data include a result for each case. It develops predictive models based on both input and output data. This kind of algorithm is used for “*classification*” and “*regression.*”

- “*Classification*” is used to assign items to a “*discrete group or class*” based on specific set of features. For example: “*tumor classification*” as “*cancerous*” or “*benign,*” classification on input data with applications

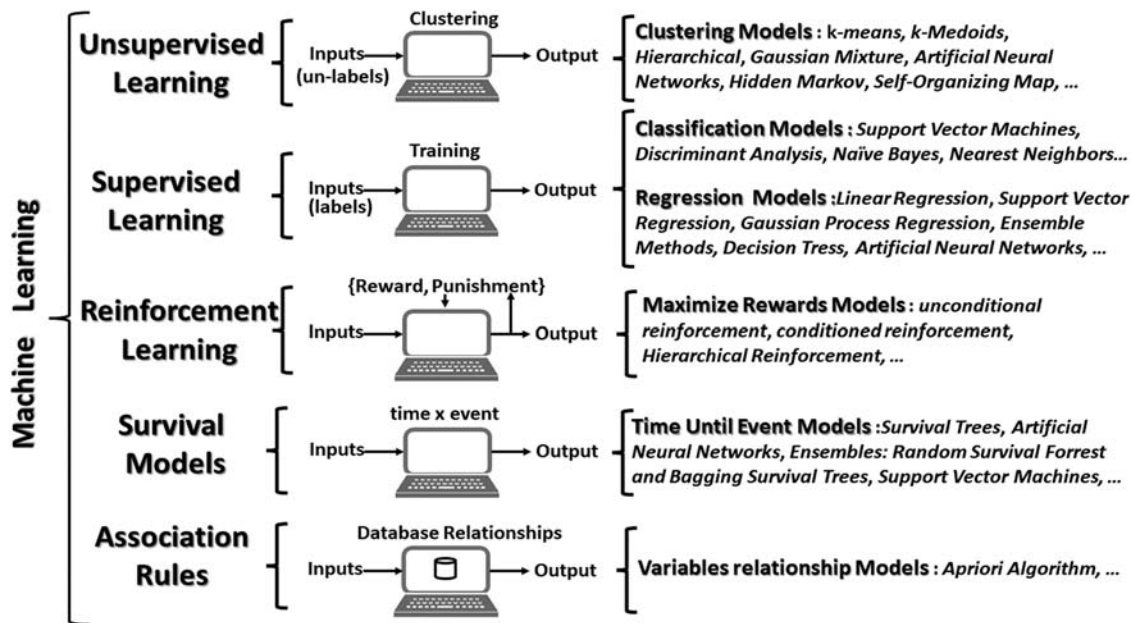


FIGURE 4.1 ML general techniques with their main characteristic, and frequently used algorithms.

for “medical imaging,” “speech recognition,” etc. The “ML models” frequently used for “Supervised Learning: Classification” are shown in Fig. 4.1, they are: “Support Vector Machines (SVM),” “Discriminant Analysis,” “Naive Bayes,” “Nearest Neighbor,” and others.

- “Regression” is used to produce “continuous responses” using a function that describes the relationship between inputs and outputs and predicts how the outputs should change as the inputs change. For example, to “predict changes on body temperature,” “predict heart attacks,” and many other predictions based on data from previous patients, such as age, weight, height, blood pressure, etc. The “ML models” frequently used as “Supervised Learning: Regression” are indicated in Fig. 4.1, they are “Linear Regression,” “Support Vector Regression (SVR),” “Gaussian Process Regression (GPR),” “Ensemble Methods,” “Decision Tress,” “Artificial Neural Networks,” and others.

4.2.3 Reinforcement learning

“Reinforcement Learning (RL)” is inspired by behaviorist psychology, concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. In others words, “RL” are “ML algorithms” that focus on training following the “attempts and failures approach” similar to “trial-and-error learning” but adding a “reward.” “RL” is considered as a “reward-based learning” consisting of an “agent” that evaluates the current situation as a “state,” takes an “action,” and receives a “reward” if is

positive or “punishment” if is negative from the “environment” in each attempt. “RL” does not use a labeled dataset as in “Supervised Learning” and the “agent” is not told which actions to follow to find a way of performing a task; it only depends on the “rewards” and “penalties” in each decision to signal that the taken option is correct or incorrect through a “feedback” that is sent when a task is completed. “The goal is to find a set of consecutives actions or sequential decision that maximize the “reward,” and each agent’s decision can affect its future actions.” “RL” analyzes data to find interconnections between data points and structures them by similarities or differences. There are two major biological categories for reinforcement: “Primary” and “Secondary”:

- “Primary reinforcement”, known also as “unconditional reinforcement”, occurs naturally in the human body without any effort for learning, such as “sleeping,” “breathing air,” “eating,” etc. “Genetics” and “previous experience” has a role in reinforcing “positive or negative feedback.”
- “Secondary reinforcement,” known also as “conditioned reinforcement,” uses a “stimulus as reward” when it is paired with another reinforcing stimulus,”- such as “training a child,” etc.

“RL” has many applications in “biomedical engineering” because it was inspired by learning rules developed in biology, and artificial agents are “sensory inputs” that define the “states” of the “environment” and the outcome of chosen actions obtain “rewards” or “punishments” that the agent tries to optimize to survive and be useful [1].

Common “*RL*” algorithms are shown in Fig. 4.1. Some “*BME*” examples are found in “*computational neuroscience*” [2,3] where the study of “*synaptic*” plasticity shows that the timescales of learning are directly related to the “*dynamical processes*” of “*neurons*” and “*synapses*” as explained in Section 2.3.1 and shown in Fig. 2.2. Different studies on animal behaviors that learn training [4], based on *Hierarchical Reinforcement Learning (HRL)*, and others methods.

Note*: Most of the “*RL algorithms*” employ “*DL*” models, which are going to be explained in Chapter 6, “*Deep Learning Models Evolution Applied to Biomedical Engineering.*”

4.2.4 Survival models

“*Survival models*” are used to analyze data in which the time until the event is of interest. The response is often referred to as “*failure time*,” “*survival time*,” or “*event time*.” “*Survival analysis*” is time to event analysis, this happens when the outcome of interest is the time until an event occurs. One important concept in “*survival analysis*” is the concept of “*censoring*,” that is, the survival times of some individuals might not be fully observed due to different reasons. This might happen when the survival study (e.g., the clinical trial) stops before the full survival times of all individuals can be observed, or a person drops out of a study, or for long-term studies, when the patient is lost to follow-up, etc. During a survival study either the individual is observed to fail at time “*T*,” or the observation on that individual ceases at time “*c*.” Then the observation is “*min(T,c)*” and an indicator variable “*I_c*” shows if the individual is censored or not. The calculations for hazard and survivor functions must be adjusted to account for censoring. The “*survival function s(t)*” indicates the probability that the individual survives for longer than a certain time “*t*” as shown in Eq. (4.1). The most commonly used distributions “*p*” are “*exponential*,” “*Weibull*,” “*lognormal*,” “*Burr*,” and “*Birnbaum–Saunders distributions*” [5].

$$\text{Survival function probability } s(t) = p(T \geq t) \quad (4.1)$$

The accumulative probability of survival as a function on time “*S(t)*” describes the probability that the survival time of an individual exceeds a certain value as shown in Eq. (4.2).

$$\text{Survival function accumulative probability } S(t) = 1 - F(t) \quad (4.2)$$

where *S(t)* is the complement of the cumulative continuous distribution function, and *F(t)* is the probability that the survival time is less than or equal to a given point of time.

The “*Death density function f(t)*” can be obtained as shown in Eq. (4.3).

$$\text{Death density function } f(t) = \frac{dF(t)}{dt} = -\frac{dS(t)}{dt} \quad (4.3)$$

And the “*Hazard function*”, which represents the probability that the event of interest occurs in the next instant given a survival time “*t*,” is shown in Eq. (4.4).

$$\text{Hazard function } h(t) = \frac{f(t)}{S(t)} = -\frac{d[\ln S(t)]}{dt} \quad (4.4)$$

“*Survival analysis*” can be made using classic “*statistical methods*,” “*ML methods*,” and others. The most frequently used “*ML models for Survival analysis*” are shown in Fig. 4.1; they are “*Survival Trees*,” “*Artificial Neural Networks*,” “*Ensembles using: Random Survival Forest and Bagging Survival Trees*,” “*Support Vector Machines (SVMs)*,” and others. Some examples in “*biomedical engineering*” are the following: “*infections: for time-to-events as the time until get infection*”; “*diseases: analysis for reoccurrence of a specific disease as breast cancer*” [6]; “*health science: analysis of time for patients’ recovery*”; and many other applications.

4.2.5 Association Rules

“*Association Rules*” is a rule-based ML method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using some measures of interestingness.

“*Association Rules*” does not extract individual preference, it finds relationships between a set of elements of every distinct case or transaction. An Association rule has “*itemset*”, this is a list of “*antecedents*” and their related “*consequents*.” “*Association Rules*” has infinity applications for “*Knowledge Discovery in Database (KDD)*.” For example, in “*Neurology*,” the “*antecedent*” could be the “*symptoms*” and the “*consequent*” could be the “*diseases*” as shown in Eq. (4.5).

$$\begin{aligned} &\text{Association Rules for Neurologic diseases example} \\ &\{ \text{Antecedents} \} \rightarrow \{ \text{Consequent} \} \\ &\{ \text{Typical motors - symptoms, Non - motors symptoms} \} \\ &\quad \rightarrow \{ \text{Neurologicdisease} \} \end{aligned}$$

$$\begin{aligned} \text{itemset} = \{ &\text{typical motors - symptoms,} \\ &\text{Non - motors symptoms, Neurologicdisease} \} \end{aligned} \quad (4.5)$$

These kinds of association are very helpful for recommending medical procedures, therapies, medicine dosages, detecting the degree of the disease, etc. Various metrics are used to understand the strength of the association between “*antecedent*” and “*consequents*,” such as “*support*,” “*confidence*,” and “*lift*”:

- “*Support*” gives a value related to the frequency of an itemset in all the cases. Mathematically, “*support*” is the fraction of the total number of transactions in which the itemset occurs, as indicated in Eq. (4.6).

Association Rules: Support metric Support

$$(\{X\} \rightarrow \{y\}) = \frac{T(\{X\}, \{Y\})}{T_T} \quad (4.6)$$

where $\{X\}$ is the list of first items, for example, typical motor symptoms; $\{Y\}$ is the list of second items, for example, typical nonmotor symptoms; $\text{Support}(\{X\} \rightarrow \{y\})$ is the frequency of $\{X\}$ associated with $\{Y\}$; $T(X,Y)$ is the number of transactions containing both $\{X\}$ and $\{Y\}$; and T_T is the total number of all kind of transactions.

- “*Confidence*” is the value of the concurrent of the “*consequent*” that has the “*antecedent*,” as indicated in Eq. (4.7).

Association Rules: Confidence metric Confidence

$$(\{X\} \rightarrow \{y\}) = \frac{T(\{X\}, \{Y\})}{T(\{X\})} \quad (4.7)$$

- “*Lift*” is a value for the “*support*” of “*consequent*” while calculating the “*conditional probability of the occurrence*” of “ $\{Y\}$ ” given “ $\{X\}$,” as indicated in Eq. (4.8).

Association Rules: Lift metric Lift

$$(\{X\} \rightarrow \{y\}) = \frac{T(\{X\}, \{Y\})/T(\{X\})}{T(\{Y\})}$$

The results are obtained generating the general rules from the entire list of items and identifying the most important ones. “*Association Rules*” is summarized as shown in Fig. 4.1. One frequently used algorithm is known as “*Apriori algorithm*” that executes a pruning to efficiently get all the frequent “*itemset*” [7,8].

There is not an easy method to choose the best “*ML model*.” To find the best model depends on the type of machine learning problem, generally this can be: “*Unsupervised Learning*,” “*Supervised Learning*,” “*Reinforcement Learning*,” “*Survival Models*,” “*Association Rules*,” and others. Each “*ML model*” type has its strengths and weaknesses for each specific data scenario. Note* = There are Automated Machine Learning that automates and eliminates manual steps required to go from a data set to a predictive models as “*AutoML from Mathworks*,” “*AutoML for IBM AI* (see research tutorial section 4.12.2.4)”, and many others.

4.3 ML clusters, classification, and regression models

Three of the most relevant tasks in “*ML*” are “*Clustering*,” “*Classification*,” and “*Regression*”:

- “*Clustering*” is defined as the partitioning of a data into subsets known as “*clusters*,” so that data in each subset ideally share some characteristics. In general, “*ML Clusters models*” are used to draw conclusions from the assigned items to a “*discrete group or class*” based on specific set of features, as their statistical properties, distance from each other’s, etc. There are two general types of “*clusters*,” the most commons are: “*Hierarchical*” and “*Partitional*.”
 - “*Hierarchical*” is a cluster tree known as a “*dendrogram*” to represent data, where each group is represented as a “*node*” linking to two or more successor groups. The groups are nested and organized as a “*tree*,” which ideally ends up as a meaningful classification scheme.
 - “*Partitional*” decomposes a dataset into a set of disjoint clusters. Given a dataset of “ N ” points, a partitioning method constructs “ K ($N \geq K$)” partitions of the data, with each partition representing a cluster.
- “*Classification*” has the objective of dividing the samples into “*classes*” and uses a training set of previously labeled data. In general, “*ML Classification models* try to draw some conclusions from the input data given for training.” There are two ways to assign a new value to a given class: “*Crispy classification*,” where for a given input the classifier returns its label; and “*Probabilistic classification*,” where for a given input the classifier returns its probabilities to belong to each class. The “*Probabilistic classification*” can be used to generate models in two ways: “*Generative models*” and “*Discriminant models*.” The “*Generative models*” learn the joint probability distribution $p(x, y)$; it predicts the conditional probability with the help of “*Bayes Theorem*.” The “*Discriminant models*” learn the conditional probability distribution “ $p(y|x)$.” There are three general classification types: “*Binary classification*,” “*Multiclass classification*,” and “*Multilabel classification*”:
 - “*Binary classification*” is a classification task with two possible outcomes, for example, “*True or False*,” “*Male or Female*.”
 - “*Multiclass classification*” is a classification task with more than two classes, each sample is assigned to one and only one target label, for example, in neurology diseases: “*Parkinson’s*,” “*Alzheimer’s*,” “*multiple sclerosis*,” etc.
 - “*Multilabel classification*” is a classification task where each sample is mapped to a set of target





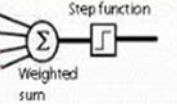
ML MODEL	Diagram	ML TYPE	Data	Hardware	Typical Applications
Naïve Bayes algorithms		Supervised Learning Probability Classifier	Simple	When CPU & memory are limited	Real time prediction, Text classification/Spam filters, recommendation of system.
k-nearest neighbor (kNN)		Supervised Learning Classification or regression or Search	Simple in small number of inputs	CPU power and memory depend of data size	Identify similar objects, "kNN" search, recommendation of system. Note: Fooled with irrelevant attributes (overcome using weights)
Decision Trees (DT)		Unsupervised or Supervised Learning Classification or regression	Simple & medium	CPU & memory according with data size	"DT" useful to share the results and how a conclusion was reached. Note: Tend to overfit (overcome using ensembles methods)
Support Vector Machine (SVM)		Supervised learning with associated learning Classification or regression	Simple & medium	CPU & memory prop. with data size & classes	Extremely accurate - no overfit Note: Time to be trained increase with number of classes greater than 2, it can be trained and tuned up front.
Artificial Neural Networks (ANN)		Unsupervised or Supervised Learning Pattern classifier	Simple, medium & large	computationally expensive	Resolve variety of problems Note: time consuming

FIGURE 4.2 Frequently used ML family models, part 1.

labels that represent more than one class, for example, about "diseases," "a person," and "stage" at the same time.

Note*: "Classification" is in some way similar to "clustering," but it requires to know ahead of time how "classes" are defined in the dataset.

- "Regression" has the main objective to obtain a model that can predict new values based on the past ones. "Regression" inferences compute the new values for a dependent variable based on the values of one or more measured attributes. In general, "ML Regressions models are used to predict assign items to a class." There are many general classification types; the more frequently used are "Linear regression," "Polynomial regression," "Logistic regression," "Ridge regression," "Support vector regression," and others, where:
 - "Linear regression" represents the relationship between the dependent variable and independent variables assumed to be linear in nature.
 - "Polynomial regression" represents the fit of a nonlinear equation by taking polynomial functions of independent variable.
 - "Logistic regression" is used when the dependent variable is binary having two categories, for example, "true or false." In "Logistic regression" the independent variables can be continuous or binary.

- "Ridge regression" is used when there exists a "collinearity" near-linear relationship among the feature variables. It is important to mention that "linear or polylinear regressions" fails when there is high "collinearity" among the feature variables.
- "Support vector regression" is used to find a function such that all points are within a certain distance from this function; it can solve both linear and nonlinear models.

Some common "ML Clusters, Classification and Regression family models" are summarized in Figs. 4.2 and 4.3, the most common are: "Naive Bayes," "K-Nearest Neighbor," "Decision Trees," "Support Vector Machine," "Artificial Neural Networks," "Discriminant analysis," "Logistic Regression Classifier," "Ensemble Classifiers," and others.

4.4 Naive Bayes family models for supervised learning

"Naive Bayes classifiers" is a family of algorithms that inherits the following attributes: "Discriminant functions," "Probabilistic generative models," "Bayesian theorem," "Naive assumptions of independence," and "Equal importance of feature vectors."

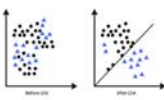
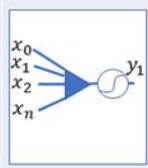

ML MODEL	Diagram	ML TYPE	Data	Hardware	Typical Applications
Discriminant analysis classifiers		Supervised Learning Pattern recognition / Classification	Simple	When CPU & memory are limited	Preprocessing step for pattern classification and ML apps. Note: minimize overfitting and computational costs.
Logistic regression classifier		Supervised Learning / Classification	large	Medium CPU & memory	On small datasets frequently is used "Naive Bayes Classifier", but as the training set size grows, get better results can be obtained with "Logistic Regression Classifier".
Ensemble classifiers		Unsupervised or Supervised Learning Classification or regression	Simple & medium	CPU & memory according with data size	Where the predictions are combination of different AI models by an algorithm to reduce "bias" and "variance".
others					

FIGURE 4.3 Frequently used ML family models, part 2.

4.4.1 Models: Gaussian Naive Bayes, multinomial Naive Bayes, Bernoulli Naive Bayes, Kernel Naive Bayes

The most common family members in "Naive Bayes classifiers" are: "Gaussian Naive Bayes" and the "Kernel Naive Bayes." They are a special type of "Supervised Learning" because a label is needed to calculate the probability for a specific value of a specific feature. These algorithms could do "simultaneous multiclass predictions." The "generative models" are based on the assumptions of the random variable mapping of each feature vector these may even be classified as "Gaussian Naive Bayes" using normal distribution, "Multinomial Naive Bayes" used for discrete counts, "Bernoulli Naive Bayes" used a binomial model for binary vectors, etc. In contrast to the "Naive Bayes operator," the "Naive Bayes Kernel operator" can be applied on numerical attributes, where the kernel is a weighting function used in nonparametric estimation techniques.

The simplest Bayesian network models are a family of simple "probabilistic classifiers" based on applying "Bayes' theorem" that indicates strong (naive) independence assumptions between the features "A" and "B," that is, how often "A" happens given that "B" happens, indicated as " $p(A|B)$," when we know how often "B" happens given that "A" happens indicated as " $p(B|A)$," and how likely "A" and "B" are on their own " $p(A)$ " or " $p(B)$," as shown in Eq. (4.8) [9].

$$\text{Bayes' theorem } p(A|B) = \frac{p(B|A)p(A)}{p(B)} \quad (4.8)$$

"Naive Bayes classifier calculates the probabilities for every factor assuming that the features are independent," for example, if we have an "event E" with "n" features $\{x_n\}$ indicated as: " $E = x_1, x_2, \dots, x_n$." We first calculate the probability for each feature given "event E" happens as: " $p(\frac{x_1}{E}), p(\frac{x_2}{E}), p(\frac{x_3}{E}), \dots, p(\frac{x_n}{E})$." Then make a selection of the feature based on the maximum probability value. In summary, this kind of "ML" known as "Naive Bayes classifiers is just a matter of counting how many times each attribute cooccurs with each class."

4.5 k-Nearest neighbor family models for supervised learning

"k-Nearest neighbor (kNN)" is a family of algorithms that categorize data points based on their distance to other points in a training dataset as a way to classify data into a "class" among its "k" nearest neighbors. They can also be used for "regression," where the output is the property value for the object.

4.5.1 Family models: fine kNN, medium kNN, coarse kNN, cosine kNN, cubic kNN, and weighted kNN

The most commonly used members in the "k-nearest neighbor family" are: "fine kNN," "medium kNN," "coarse kNN," "cosine kNN," "cubic kNN," and "weighted kNN." In these algorithms there are no training phases.

The “*kNN*” family of classification algorithms and regression algorithms is often referred to as “*memory-based learning*” or “*instance-based learning*” or “*lazy learning*.” “*kNN*” belongs to the family of “*supervised ML*” algorithms, which means we use a “*labeled identifier*” in the “*target variable*” dataset to predict the “*class*” of new data point. A variety of distance criteria can be chosen from: the “*kNN* algorithm” gives the user the flexibility to choose distance while building the “*kNN* model.” Based on:

- “*Euclidean distance*” ordinary straight-line distance between two points in Euclidean space, as shown in Eq. (4.9)

$$\text{Euclidian distance } \|a-b\|_2 = \sqrt{\sum_i^n (a_i - b_i)^2} \quad (4.9)$$

- “*Hamming distance*” is a metric for measuring the edit distance between two sequences, as shown in Eq. (4.10)

$$\text{Hamming distance or Squared Euclidian distance} \quad (4.10)$$

$$\|a-b\|_2^2 = \sum_i^n (a_i - b_i)^2$$

- “*Manhattan distance*” is the sum of the lengths of the projections of the line segment between the points onto the coordinate axes, as shown in Eq. (4.11)

$$\text{Manhatan distance } \|a-b\|_1 = \sum_i^n |a_i - b_i| \quad (4.11)$$

- “*Minkowski distance*” is a metric in a normed vector space which can be considered as a generalization of both the “*Euclidean distance*” and the “*Manhattan distance*”.

The “*k-nearest neighbor (kNN) family models*” are considered in general as “*crisp classification algorithms*” with the exception of the “*fuzzy KNN classification algorithms*” [10].

In summary “*kNN* assumes that similar things exist in close proximity.” In other words, similar things are near to each other. The steps are as follows:

- Load the data, that is, a pair of vectors. $[X_n, Y_n]$, where n is the number of elements, X_n is the input vector, and Y_n is the output class.
- Initialize “ k ” to a chosen number of neighbors.
- For each example in the data:
 - Calculate the distance between the query example and the current example from the data;

- Add the distance and the index of the example to an ordered collection;
- Sort the ordered collection of distances and indices from smallest to largest by the distances;
- Pick the first “ k ” entries from the sorted collection;
- Get the labels of the selected “ k ” entries;
- If regression, return the mean of the “ k ” labels;
- If classification, return the mode of the “ k ” labels.

“*kNN*” has the following advantages: robust, intuitive, and simple algorithms; does not take many assumptions; does not need training steps; constantly evolves; can also be implemented for multiclass problems; for “*classification*” and “*regression*,” one needs only one “*hyperparameter*” as the parameter whose value is set before the learning process begins, and the rest of the parameters are aligned to it; and it is flexible offering various methods to calculate distances. “*kNN*” has the following disadvantages: slow algorithm because it works well with small number of input variables but is very slow when the number of variables increases; it is hard to choose the optimal number of neighbors while classifying new data; it can be fooled by irrelevant attributes that obscure important attributes unless data weights are applied; in addition it needs all values.

4.6 Decision trees family models for supervised learning

“*Decision Trees (DT)*” are a family of algorithms that split the original dataset into two or more subsets at each algorithm step, until isolating the desired “*classes*.” Each step produces a split in the dataset and each split can be graphically represented as a node. “*DT*” algorithms solve “*binary or multinomial classification*” problems and can be used as: “*classifiers*” to obtain nominal responses (“*true*” or “*false*”) and “*regression*” to obtain numeric responses.

4.6.1 Family models: fine decision tree, medium decision tree, and coarse decision tree

The most frequently used “*Decision Trees*” family members are: “*Fine Decision Tree*,” “*Medium Decision Tree*,” and “*Coarse Decision Tree*.” In “*Decision Trees classifier*,” one way to display this algorithm is through conditional control statements known as “*learning decision rules from features*.” To split the nodes at the most important features an “*objective impurity function (OI)*” must be defined, as shown in Eq. (4.12), that maximizes the information gain in each tree split [10].

Decision Trees Classifier objective impurity function

$$OI(D_p, f) = I(D_p) - \left(\frac{N_{left}}{N_p} I(D_{left}) + \frac{N_{right}}{N_p} I(D_{right}) \right) \quad (4.12)$$

where f is the feature to perform the split, D_p is the dataset of the parent, I is the impurity measure, D_{left} and D_{right} are the dataset for the child nodes, and N_{left} and N_{right} are the number of samples in the child nodes.

In “*Decision Trees regression*” the “*impurity measurement function*” is defined using “*weighted mean squared error (MSE)*,” as shown in Eq. (4.13).

Decision Trees Regression objective impurity function

$$\text{Weighted mean squared error } MSE(t) = \frac{1}{N_t} \sum_{i \in D_t} (y^i - \hat{y}_t)^2 \quad (4.13)$$

where N_t is the number of training samples at node t , D_t is the trainer subset at node t , y^i is the true target value, and $\hat{y}_t = \frac{1}{N_t} \sum_{i \in D_t} (y^i)$ is the predicted target value based on the sample mean.

“*DT*” is relatively quick and easy to follow; it shows a full representation of the path taken from root to leaf. This is especially useful if you need to share the results with people interested in how a conclusion was reached. The main disadvantage of “*decision trees*” is that they tend to “*overfit*,” but there are “*ensemble methods*” to counteract this [11]. The “*Decision Trees (DT) family models*” are considered in general as “*crisp classification algorithms*” with the exception of the “*fuzzy KNN classification algorithms*” [12].

In summary, “*DT*” is a decision support tool that uses a “*tree-like graph*” or “*model of decisions*” and their possible consequences, including chance event outcomes, resource costs, and utility. The “*DT*” process is explained in the following steps:

- It begins with a “*Root node*” that represents the entire population or sample.
- Then split the node into two or more subnodes. If a subnode can be split further, it is known as a “*decision node*,” if the subnode cannot be split it is called a “*terminal node*.”
- A subsection of the entire tree is called a “*branch or subtree*.”
- A node that is divided into subnodes is called the “*parent node*” of subnodes where each subnode is the child of a parent node.
- When a subnode is removed from a “*decision node*,” the process is called “*pruning*.”

4.7 Support vector machine family members

“*Support Vector Machine (SVM)*” is a family of algorithms for generalized “*Linear Classifiers and Regression*” analysis as an extension of the “*perceptron*,” that is a computer model that simulates the ability of the human brain to recognize and discriminate.

4.7.1 Family models: linear SVM, fine Gaussian SVM, medium Gaussian SVM, coarse Gaussian SVM, quadratic SVM, and cubic SVM

The most commonly members used in the “*SVM family*” are: “*linear SVM*,” “*fine Gaussian SVM*,” “*medium Gaussian SVM*,” “*coarse Gaussian SVM*,” “*quadratic SVM*,” and “*cubic SVM*.” The “*SVM*” is a “*nonprobabilistic binary linear classifier*,” except for methods such as “*Platt scaling*” that uses “*SVM*” in a “*probabilistic classification*.” When there are two classes “*SVM classifiers*” data by finding the best “*hyperplane*” that separates all data points from one class from the other class. If there are more than two classes “*SVM*” creates a set of “*binary classification*” subproblems with one “*SVM learner*” for each one. The dimension of the “*hyperplane*” depends upon the number of features. If the number of input features is 2, then the “*hyperplane*” is just a line. If the number of input features is 3, then the “*hyperplane*” becomes a two-dimensional plane. Besides, it difficult to imagine when the number of features exceeds 3. In the “*SVM algorithm the objective is maximizing the margin between the data points and the hyperplane*.” The “*loss function*” helping to maximize the margin is “*hinge loss*,” as shown in Eq. (4.14), and for a “*Linear Support Vector Machine*” as shown in Eq. (4.15) [13].

Support Vector Machine hinge loss (loss function)

$$L(y) = \max(0, 1 - t \cdot y) \text{ for } t = \pm 1 \quad (4.14)$$

where y is the raw output of the classification score not the predicted class label.

$$\text{Linear Support Vector Machine } y = w \cdot x + b \quad (4.15)$$

where (w, b) are the parameters for the hyperplane and x is the input variable (s).

In Eq. (4.14) when, “ t ” and “ y ” have the same sign, if “ y ” predicts the correct “ $|y| \geq 1$ ” then the “*hinge loss*” $L(y) = 0$. But when “ t ” and “ y ” have the different sign: “*hinge loss* $L(y)$ ” increases linearly with “ y ,” and similarly if “ $|y| < 1$ ” even if it has the same sign for the correct prediction it indicates that there is not enough margin. “*SVM*” needs to be trained and tuned up front, an investment of time is needed in the model before

beginning to use it. Also, its speed is heavily impacted if we are using the model with more than two classes. The “SVM family models” are considered in general as “*crisp classification algorithms*,” except for the “*fuzzy SVM classification algorithms*.”

The “SVM” is fast replacing “ML” using basic “ANNs” for some applications, such as being the tool of choice for prediction and pattern recognition tasks, primarily due to its ability to generalize well on “unseen data.”

4.8 Artificial neural network family models

“Artificial Neural Networks (ANNs)” are a family that define individual computing elements that are connected and by the strengths of those connections based on “weights” calculations. The “weights” are automatically adjusted by training the network according to a specified learning rule until it performs the desired task correctly. “ANNs” are based on the “perceptron,” that is a mathematical model of a biological “neuron.” As explained in Section 1.4.10 and illustrated in Fig. 1.6; in actual “neurons” the “dendrite” receives electrical signals from the “axons” of other “neurons,” in the “perceptron model” these electrical signals are represented as numerical values. At the “synapses” between the “dendrite” and “axons,” electrical signals are modulated in by various amounts. This is also modeled in the “perceptron” by multiplying each input value by a value called the “weight.” A “neuron” fires an output signal only when the total strength of the input signals exceeds a certain “threshold.” We model this phenomenon in a perceptron by calculating the “weighted sum of the inputs” to represent the total strength of the input signals, and applying a “step function” on the sum to determine its output. As in “biological neural networks,” this output is fed to other “perceptrons.” They perform tasks by considering examples, generally without being programmed with task-specific rules. “ANNs” are great at modeling nonlinear data with a high number of input features. When used correctly, “ANNs” can solve problems that are too difficult to address with a straightforward algorithm. However, “neural networks” are computationally expensive, it is difficult to understand how they reach a solution and infer an algorithm, and fine-tuning an “ANN” is often unpractical because only the number of inputs can be adjusted in the training setup and retrain.

“ANNs” models are very flexible and can be obtained from methods, such as “nonprobabilistic” or “probabilistic model.”

- “Nonprobabilistic” is applied to “pattern recognition” and it has been demonstrated in many studies that they are capable of providing accurate results and reliable identification [14].
- “Probabilistic model” is applied in “ANN” models such as “Probabilistic Neural Network (PNN)”, which is a “feed forward neural network,” that is widely used in “classification and pattern recognition” problems. In the “PNN algorithm,” the parent probability “distribution function (PDF)” of each class is approximated by a “Parzen window” and a “nonparametric function” [14].

Note: “Probabilistic generative” methods learn the posterior class probabilities explicitly. As opposed to it, “Probabilistic discriminative” methods learn the posterior class probabilities directly.

They can be used in “Unsupervised or Supervised learning”:

- “Unsupervised learning” is where the targets are not specified and “ANNs” can be used for “patterns classifications” with the goal of group similar units within certain ranges.
- “Supervised learning” is where the target patterns are given in form of binary values of the decimal numbers. In the learning, some input patterns are propagated through the net that can have different structures until reaching the output layer where they are compared with the target pattern and an error value is computed, for greater values the “weights” values in each node will be adjusted in the next calculation loop.

In “ANNs” the next calculation loop can generally use: “feed forward,” “backpropagation,” “recurrent,” “memory augmented,” “memory augmented,” “modular,” and “evolutionary,” as indicated in Fig. 1.7.

4.8.1 Feed forward neural network family models: perceptron, multilayer perceptron, radial basis network, probabilistic neural network, extreme learning machine

“Feed Forward Neural Network” implies a signal that can only be fed forward, meaning the absence of recurrent or feedback connections. In other words, the data path in the network is only forward facing, no backward feed connections between neurons are present, as indicated in Fig. 1.8.

“Feed forward neural network” families are studied in more detail and with examples in Section 5.2.

4.8.2 Backpropagation neural networks

“Backpropagation neural networks” imply that the signal propagates from the input data forward through its parameters toward the decision, and then propagates information about the error in reverse, so in this way it can adjust the parameter until finding the smallest error. Some frequently used examples of Backpropagation neural networks were shown in Fig. 1.9, these are “Auto Encoder (AE),” “Variational Auto Encoder (VAE),” “Denoising Auto Encoder (DAE),” “Sparse Auto Encoder (SAE),” and others.

“Backpropagation neural networks” families are studied in Section 5.4.

4.9 Discriminant analysis family models

“Discriminant analysis” is a technique used to analyze research data when the criterion or the dependent variables are “categorical” and the predictor or the independent variable is interval in nature.

4.9.1 Family models: linear discriminant analysis, quadratic discriminant analysis

“Discriminant analysis” is a “classifier” that finds “pattern recognition” of a linear combination of features to separate two or more “classes.” “Discriminant analysis” is a “supervised learning” that assumes that different classes generate data based on different “Gaussian distributions.” For training the “classifier” is a “fitting function” that does not use prior probabilities or cost for fitting, estimating the parameters of a “Gaussian distribution” for each “class.” To predict the classes of new data the trained classifier finds the “class” with the smallest misclassification cost. The most frequently used methods for “Discriminant analysis classifiers” are “Linear” and “Quadratic:

- “Linear Discriminant Analysis (LDA)” is a “dimensionality reduction technique” because it reduces the number of dimensions (i.e., variables) in a dataset while retaining as much information as possible., it computes the “sample mean of each class.” Then it computes the “sample covariance” by first subtracting the sample mean of each class from the observations of that class, and taking the empirical covariance matrix of the result. “LDA” is closely related to “Analysis of Variance (ANOVA)” and

“regression analysis,” which also attempt to express one dependent variable as a linear combination of other features or measurements.

- “Quadratic Discriminant Analysis (QDA)” is closely related to “LDA,” where it is assumed that the measurements from each class are “normally distributed.” However, in “QDA” there is no assumption that the covariance of each of the classes is identical.

“LDA” and “QDA” can be derived from a probabilistic model of the class conditional distribution of the data for each class. The prediction can be obtained by using the “Bayes’ rule” as indicated in Eq. (4.16) and selecting the class that maximizes the conditional probability.

Discriminant analysis conditional distribution of the data

$$P(y = k|x) = \frac{P(X|y=k)P(y = k)}{P(X)} \quad (4.16)$$

where k is each class.

“Discriminant analysis” has many applications in different fields, such as [15] “medical,” “face recognition,” “robots,” “speech recognition,” “microarray data classification,” “image retrieval,” “bioinformatics,” “biometrics,” etc. For example:

- “Discriminant analysis for medical applications” is used for the classification for the state of patients’ diseases as “mild,” “moderate,” or “severe” based on the various parameters and the medical treatment the patient is going through in order to decrease the movement of treatment.
- “Discriminant analysis for face recognition” is the most famous application in the field of “computer vision.” Every face is drawn with a large number of pixel values, “LDA” reduces the number of features to a more controllable number first before implementing the classification task.
- “Discriminant analysis in robots”: for example, robots are trained to learn and talk to work as human beings; this can be treated as “classification” problems. “LDA” makes similar groups based on various parameters such as frequencies, pitches, sounds, tunes, etc.

In summary “Discriminant analysis (DA)” is a preprocessing step for “pattern classification” and “ML applications” used for dimensionality reduction in multivariate datasets. It projects the dataset into moderate dimensional space with a genuine class of separable features that minimize overfitting and computational costs.

4.10 Logistic regression classifier

“*Logistic regression classifier*” is a “*statistical learning*” technique that uses a “*linear regression*” to produce discrete binary outputs that attempt to maximize the quality of the output on a training set.

4.10.1 Family models “*logistic regression*”

“*Logistic regression*” is categorized as “*ML Supervised Learning*” for “*classification*.” The inputs are continuous features-vectors or matrices “*X*,” as shown in Eq. (4.17a), containing a number of features. The inputs enter an activation function known as “*Sigmoid/logistic function sig(t)*,” as shown in Eq. (4.17b), and the outputs are in a discrete vector “*Y*” as a binary variable with values {0,1}, as shown in Eq. (4.17c), that we can assume is “*Bernoulli distributed*” with probability parameters.

Logistic Regression Classifier components

$$\text{Input } X(n,k) = \begin{bmatrix} x_{1,1} & \dots & x_{1,k} \\ \dots & \dots & \dots \\ x_{n,1} & \dots & x_{n,k} \end{bmatrix} \quad (4.17a)$$

$$\text{Activation function } \text{sig}(t) = \frac{1}{1 + e^{-t}} \quad (4.17b)$$

$$\text{Output } Y(n) = \begin{bmatrix} y_1 \\ \dots \\ y_n \end{bmatrix} \quad (4.17c)$$

“*Logistic regression*” has the objective function known as “*Maximum Likelihood Estimation (MLE)*” as indicated in Eq. (4.18).

$$\text{Logistic Regression Classifier MLE function} \\ \arg \max_{\beta} : \log \left\{ \prod_{i=1}^n P(y_i|x_i)^{y_i} (1 - P(y_i|x_i))^{(1-y_i)} \right\} \quad (4.18)$$

where y is the output with values {0,1}, $P(y_i|x_i)$ is the posterior probability, and β is the vector of “*weights/coefficients*.”

“*Logistic regression*” is used to describe data and to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval, or ratio-level independent variables. “*Logistic regression*” in BME has many applications, such as the “*prediction of diseases such as tuberculosis and others*” [15], “*analysis of genes for cancer research*” [16], and many more.

In general, on small datasets “*Naive Bayes classifier*” is frequently used, but as the training set size grows, better results can be obtained with “*Logistic regression classifier*.” The reason is that the “*ML discriminative models*” use linear equations as building blocks and attempt to maximize the quality of output on a training center, such as “*Support vector machines*,” “*Logistic regression classifier*,” and

(Continued)

“*Perceptron*.” Otherwise, “*Naive Bayes classifiers*” are “*ML generative classifiers*” that learn a model of joint probabilities “ $p(x, y)$ ” and use the “*Bayes’ theorem*” to make a rule to calculate the prediction. In classification, “*Naive Bayes*” converges quicker but has typically a higher error than “*Logistic regression*” which is a “*probabilistic discriminative method*.”

4.11 Ensemble classifiers family models

“*Ensemble classifiers*” consist of a set of individually “*ML trained classifiers*” where the predictions are combined by an algorithm to reduce “*bias*” and “*variance*.”

4.11.1 Models: AdaBoost, RUSBoost, Subspace kNN, Random Forrest, Subspace discriminant

“*Ensemble classifiers*” use “*Supervised learning*.” Some frequently general techniques applied in “*Ensemble classifiers*” are [17] “*Stacking*,” “*Blending*,” and “*Bagging*”:

- “*Stacking*” is when a single training dataset of size “ m ” is given to multiple models “ n ” and trained in parallel. The training set is further divided using “*k-fold validation*” and each resultant model “ M ” is created from different algorithm. The predictions from the “ M models” are used as “*predictors*” for the final classification model with more accuracy.
- “*Blending*” is when the dataset is directly divided into “*training*” and “*validation*” instead of the “*k-fold validation*” then continuing the same procedure described in “*Stacking*.”
- “*Bagging*” uses “ n ” sampling of training data generated by randomly picking various data items instead of the “*k-fold validation*” then continuing the same procedure described in “*Stacking*.”

“*Ensemble classifiers*” use techniques based on “*Stacking*,” “*Blending*,” or “*Bagging*” by following the next three steps in each iteration:

1. A base model is created for each classifier algorithm to uses;
2. Models runs in parallel and independent of each other; and
3. The final predictions are determined by combining the predictions from all the models. After the evaluation of each model, the misclassified data are given more weight so that the next model has more focus on these items. The final model is averaged as indicated in Eq. (4.19).

“Stacking/Blending/Bagging Ensemble classifiers”

final average model

$$e = \frac{\sum_{i=1}^{i=n} e_i}{n} \quad (4.19)$$

where base classifiers are e_1, \dots, e_n , n is number of models, and e is final classifier.

Another frequently used technique in “Ensemble classifiers” is “Boosting,” which is a self-learning technique that apply “weights calculation.” It initially starts with equal “weights” for the first model, then the resultant predictions are assigned a “weight based on its performance” for the next model. After the evaluation of each model, the misclassified data are given more “weight” so that the next model has more focus on these items. In summary, “Ensemble classifiers” using “Boosting” follow the next steps in each iteration:

1. “Weight” each training result by how incorrectly it was classified;
2. “Reclassify” using the assigned weights; and
3. “Reweight” the results. The final model is the result from all models used that focus on various groups of data voted using their assigned weights, the final model is averaged by using the *weighted average method* as shown in Eq. (4.20).

kern “Boosting Ensemble classifiers”

Weighted Average Method

$$e = \frac{(\sum_{i=1}^{i=n} e_i w_i)}{(\sum_{i=1}^{i=n} w_i)} \quad (4.20)$$

where base classifiers are e_1, \dots, e_n weights are w_1, \dots, w_n , n is number of models, and e is final classifier.

The most frequently used variations in “Ensemble classifiers” are: “Boosted Trees” as the “AdaBoost” and the “RUSBoost,” and “Bagged trees” as “Subspace kNN,” “Random Forest,” and “Subspace discriminant”:

- “AdaBoost” takes output of the other weaker learning algorithms and is combined into a weighted sum that represents the final output of the “boosted classifier.”
- “RUSBoost” is a “Boosted tree” that can be used as “Binary and Multiclass classification” with class imbalance.
- “Subspace kNN” is known also as “Nearest neighbors in random subspaces.” It can be used as “Binary and Multiclass classification” using “Bagged trees.”
- “Random forests” or “random decision forests” are an ensemble learning method for “classification, regression, and other tasks” that operate by constructing a multitude of “decision trees” at training time and

outputting the class that is the mode of the “classes for classification” or “mean prediction for regression of the individual trees” [18].

- “Subspace discriminant” is a “Bagged tree” that has the objective of finding optimal projection vectors by simultaneously minimizing the within-class distance and maximizing the between-class distance in the projection space and optimal projection vectors can be achieved by solving a generalized eigenvalue problem. It was inspired by the idea of the “maximum margin criterion (MMC)” embedded into the eigen-subspace corresponding to the degenerated eigenvalue to exploit discriminability of the eigenvectors in the eigen-subspace.

In summary, in “Stacking/Blending/Bagging Ensemble classifiers” the models run in parallel and are independent of each other, whereas in “Boosting Ensemble classifiers” the models run in sequence and depend on the previous models.

4.12 IBM ML Solution: IBM Watson SPSS

As explained in Section 3.4, the “SPSS Modeler flows” from “IBM Watson Studio” are “AI tools” to develop “predictive models and deploy them to improve decision-making.” The flow interface supports the “data mining visual modeling” process integrating algorithms from “AI,” “ML,” and “Statistics.” It has many methods available on the “node palette” using the “flow editor,” as shown in Fig. 3.15, where we focused on the menu options: “import,” “record operations,” “field operations,” and “graphs.” In this section, we focus on the menu options: “Modeling,” “Outputs,” and “Exports” that allow one to derive new information from the data to develop predictive models, obtain “AI models,” and deploy them.

4.12.1 SPSS Modeler flows > Modeling

In the “SPSS Modeler flows” the Submenu: “Import,” “Record Operations,” “Field Operations,” and “Graph” were shown in Fig. 3.15 and studied in the last chapter. The submenu option for “Modeling,” that shows “ML models” currently available are shown in Fig. 4.4 and these are:

- “Auto Classifier*”: Node estimate and compares discrete models of nominal or binary classification using different methods.

Note*: See “Auto Classifier”: applied on: *Research tutorial example 4.3, section 4.12.2.3 IBM Watson SPSS Modeler Flow for “Kidney disease.”*

IBM Watson Studio - SPSS Modeler Flower Menu Nodes for Step 2) Build and Validate AI Models Modeling

<ul style="list-style-type: none"> Auto Classifier: Node estimate & compare discrete models of nominal or binary classification using different methods Auto Numeric: Node estimate & compare continuous numeric ranges models different methods for prediction Auto Cluster: Node estimate & compare clustering models to identify groups of records with similar characteristics Bayes Net: Probability model combine observed & recorded evidence to establish likelihood of occurrences C5.0: Algorithm to build either a decision tree or a rule set split the sample based on field provide maximum info gain C&R Tree: Classification and Regression Tree for classification & prediction, using recursive partition to split CHAID: Chi-squared Automatic Interaction Detection for classification building decision trees Quest: Quick, Unbiased, Efficient Statistical Tree for binary classification tree analysis Tree-AS: Build decision trees with CHAID or Exhaustive CHAID model Random Trees: Build an ensemble model that consists of multiple decision trees Random Forrest: Advanced implementation of a bagging algorithm with a tree model as the base mode Decision List: Identify subgroups or segments that show a higher or lower likelihood of a binary outcome Times Series: Choose estimate and build exponential smoothing, ARIMA, or multivariate ARIMA forecasts model TCM: create a temporal causal model that attempts to discover key causal relationships in time series GenLin: Expands the general linear model, dependent variable is linearly related to factors and covariates link GLMM: creates a generalized linear mixed model GLE: Identifies dependent variable as is linearly related to the factors and covariates via a specified link function. 	<ul style="list-style-type: none"> Linear: Linear regression statistical technique for classifying records based on the values of numeric input Linear-AS: Linear regression fits a straight line/surface that minimizes discrepancies between predicted & actual output Regression: Linear regression fits a straight line/surface that minimizes discrepancies between predicted & actual output LSVM: Linear support vector machine to classify data. It is suited for use with wide large number of predictor fields Logistic: also known as nominal regression, it is statistical technique classify records based on values of input fields Neural NET: It approximate a wide range of predictive model minimal demands on model structure and assumption KNN: Nearest Neighbor Analysis is a method for classifying cases based on their similarity to other cases Cox: Cox Regression builds a predictive model for time-to-event data PCA/Factor: provides powerful data-reduction techniques to reduce the complexity of the data SVM: Support vector machine to classify data, It is suited for use with wide large number of predictor fields Feature Selection: It is algorithm can be used to identify the fields that are most important for a given analysis Discriminant: Discriminant analysis builds a predictive model for group membership. SLRM: Self-Learning Response Model (SLRM) build a model continually update as a dataset grows STP: Spatio-Temporal Prediction apps for energy, buildings facilities, management, performance and forecasting Association Rules: Associate a particular conclusion with a set of conditions Apriori: It discovers association rules in the data Carma: uses association rules discovery algorithm to discover association rules in the data 	<ul style="list-style-type: none"> Sequence: Discovers patterns in sequential or time-oriented data Kohonen: Neural network that perform clustering, also known as a <i>knet</i> or a <i>self-organizing map(SOM)</i> Anomaly: Models are used to identify outliers, or unusual cases in the dataset K-Mean: Provides a method of cluster analysis, It does not use a target field. It is unsupervised learning TwoStep: Cluster analysis, not use a target field. Records are grouped within a group or cluster tend to be similar TwoStep-AS: Exploratory tool is designed to reveal natural clusters within a dataset Isotonic-AS: They are regression algorithms. The Isotonic-AS node in Watson Studio is implemented in Spark K-Means-AS: Clustering algorithms. It clusters data points into a predefined number of clusters. KDE-Modeling: Kernel Density Estimation uses the Ball Tree or KD Tree algorithms for efficient queries. Gaussian Mixture: Probabilistic model represent normally distributed subpopulations within an overall population XGBoost-AS: Learn weak classifiers and then add them to a final strong classifier. XGBoost-Tree: Advanced implementation of a gradient boosting algorithm, iteratively learn classifiers XGBoost-Linear: Implementation of a gradient boosting algorithm with a linear model as the base model. One-Class SVM: The node can be used for novelty detection. Unsupervised learning algorithm. Multilayer Perceptron: Classifier based on feedforward artificial neural network with multiple layers HDSCAN: Hierarchical Density-Based Spatial Clustering uses unsupervised learning to find clusters/dense regions Extension Model: Allows enables other models
--	---	--

FIGURE 4.4 Currently IBM Watson SPSS for submenu: *Modeling*. Note: Please review Fig. 3.15 for “IBM Watson SPSS submenu”: “Import,” “Record Operations,” “Field Operations,” and “Graph” to complement this figure.

- “Auto Numeric”: Node estimate and compares continuous numeric ranges models different methods for prediction.
- “Auto Cluster”: Node estimate and compares clustering models to identify groups of records with similar characteristics.
- “Bayes Net”: Probability model combines observed and recorded evidence to establish likelihood of occurrences.
- “C5.0”*: Algorithm to build either a decision tree or a rule set split the sample based on field provide maximum info gain.
- “C&R Tree”*: Classification and Regression Tree for “classification and prediction,” using recursive partition to split.

Note*: See “C5.0” and “C&R Tree”: applied on: *Research tutorial example 4.2, section 4.12.2.2 IBM Watson SPSS Modeler Flow for “Heart disease ML Model and deployment.”*

- “CHAID”: “Chi-squared Automatic Interaction Detection” for classification building decision trees.
- “Quest”: Quick, unbiased, efficient statistical tree for “binary classification tree analysis.”
- “Tree-AS”: Build “decision trees” with “CHAID or Exhaustive CHAID model.”

- “Random Trees”: Build an ensemble model that consists of “multiple decision trees.”
- “Random Forrest”*: Advanced implementation of a bagging algorithm with a “tree model” as the base mode.

Note*: See “Random Forrest”: applied on: *Research tutorial example 4.83, section 4.12.2.3 IBM Watson SPSS Modeler Flow for “Kidney disease.”*

- “Decision List”: Identify subgroups or “segments” that show a higher or lower likelihood of a binary outcome.
- “Times Series”: Choose to estimate and build exponential smoothing, “ARIMA,” or “multivariate ARIMA forecasts model.”
- “TCM”: Create a temporal causal model that attempts to discover key causal relationships in time series.
- “GenLin”: Expands the general linear model; dependent variable is linearly related to factors and covariates links.
- “GLMM”: Creates a generalized linear mixed model.
- “GLE”: Identifies dependent variable as linearly related to the factors and covariates via a specified link function.
- “Linear”: Linear “regression” statistical technique for “classifying” records based on the values of numeric input fields.

- “*Linear-AS*”: Linear regression fits a straight line/surface that minimizes discrepancies between predicted and actual output.
- “*Regression*”: Linear regression fits a straight line/surface that minimizes discrepancies between predicted and actual output.
- “*LSVM**”: Linear support vector machine to *classify* data. It is suited for use with large number of predictor fields.

Note*: See “*LSVM*”: applied on: *Research tutorial example 4.3, section 4.12.2.3 IBM Watson SPSS Modeler Flow for “Kidney disease.”*

- “*Logistic*”: also known as “*nominal regression*,” it is statistical technique to classify records based on values of input fields.
- “*Neural NET*”: It approximates a wide range of “*predictive models*” with minimal demands on model structure and assumption.
- “*KNN*”: k-Nearest neighbor analysis is a method for “*classifying*” cases based on their similarity to other cases.
- “*Cox*”: Cox Regression builds a “*predictive model*” for time-to-event data.
- “*PCA/Factor*”: Provides powerful “*data-reduction*” techniques to reduce the complexity of the data.
- “*SVM*”: Support vector machine “*classifies data*”; it is suited for use with a large number of predictor fields.
- “*Feature Selection*”: It is an algorithm that can be used to “*identify the fields that are most important for a given analysis.*”
- “*Discriminant*”: Discriminant analysis builds a “*predictive model*” for group membership.
- “*SLRM*”: Self-learning response model (SLRM) builds a model that “*continually updates*” as a dataset grows.
- “*STP*”: Spatiotemporal prediction apps for energy, buildings facilities, management, performance, and forecasting.
- “*Association Rules*”: Associate a particular conclusion with a set of conditions.
- “*Apriori*”: It “*discovers association rules*” in the data.
- “*Carma*”: It uses an “*association rules discovery algorithm to discover association rules in the data.*”
- “*Sequence*”: “*Discovers patterns in sequential or time-oriented data.*”
- “*Kohonen*”: “*Neural network that performs clustering*,” also known as a “*knet*” or a “*self-organizing map (SOM).*”
- “*Anomaly*”: Models are used to “*identify outliers*,” or unusual cases in the dataset.
- “*K-Means**”: Provides a method of “*cluster analysis*,” it does not use a target field. It is unsupervised learning.

Note*: See “*k-means*” applied on: *Research 4.61, section 4.12.2.1 Tutorial example IBM Watson SPSS Modeler Flow for “ML Model for Diabetes.”*

- “*TwoStep*”: “*Cluster analysis*”; does not use a target field. Records grouped within a group or cluster tend to be similar.
- “*TwoStep-AS*”: “*Exploratory tool*” designed to reveal natural clusters within a dataset.
- “*Isotonic-AS*”: They are “*regression algorithms.*” The Isotonic-AS node in Watson Studio is implemented in Spark.
- “*K-Means-AS*”: “*Clustering algorithms.*” It clusters data points into a predefined number of clusters.
- “*KDE-Modeling*”: Kernel density estimation uses the Ball Tree or KD Tree algorithms for efficient “*queries.*”
- “*Gaussian Mixture*”: “*Probabilistic model*” represents normally distributed subpopulations within an overall population.
- “*XGBoost-AS*”: Learns weak “*classifiers*” and then adds them to a final strong classifier.
- “*XGBoost-Tree**”: Advanced implementation of a gradient boosting algorithm, iteratively learns classifiers.

Note*: See “*XGBoost-Tree*”: applied on: *Research tutorial example 4.3, section 4.12.2.3 IBM Watson SPSS Modeler Flow for “Kidney disease.”*

- “*XGBoost-Linear*”: Implementation of a gradient boosting algorithm with a linear model as the base model.
- “*One-Class SVM*”: The node can be used for “*novelty detection.*” Unsupervised learning algorithm.
- “*MultilayerPerceptron*”: “*Classifier*” based on feed forward artificial neural network with multiple layers.
- “*HDBSCAN*”: Hierarchical density-based spatial “*clustering*” uses unsupervised learning to find clusters/dense regions.
- “*Extension Model*”: Allows and enables other models as plug-ins.

4.12.2 SPSS Modeler flows > Output

The “*SPSS Modeler flows*” submenus for “*Output*” and “*Export*” are shown in [Fig. 4.5](#).

“*Outputs*” menu options provide the means to obtain information about your data and models:

- “*Table*”: Creates a table that lists the values in the dataset for easy inspection or export them.
- “*Matrix*”: Creates a table that shows relationships between fields. Shows relationship between two categorical fields.

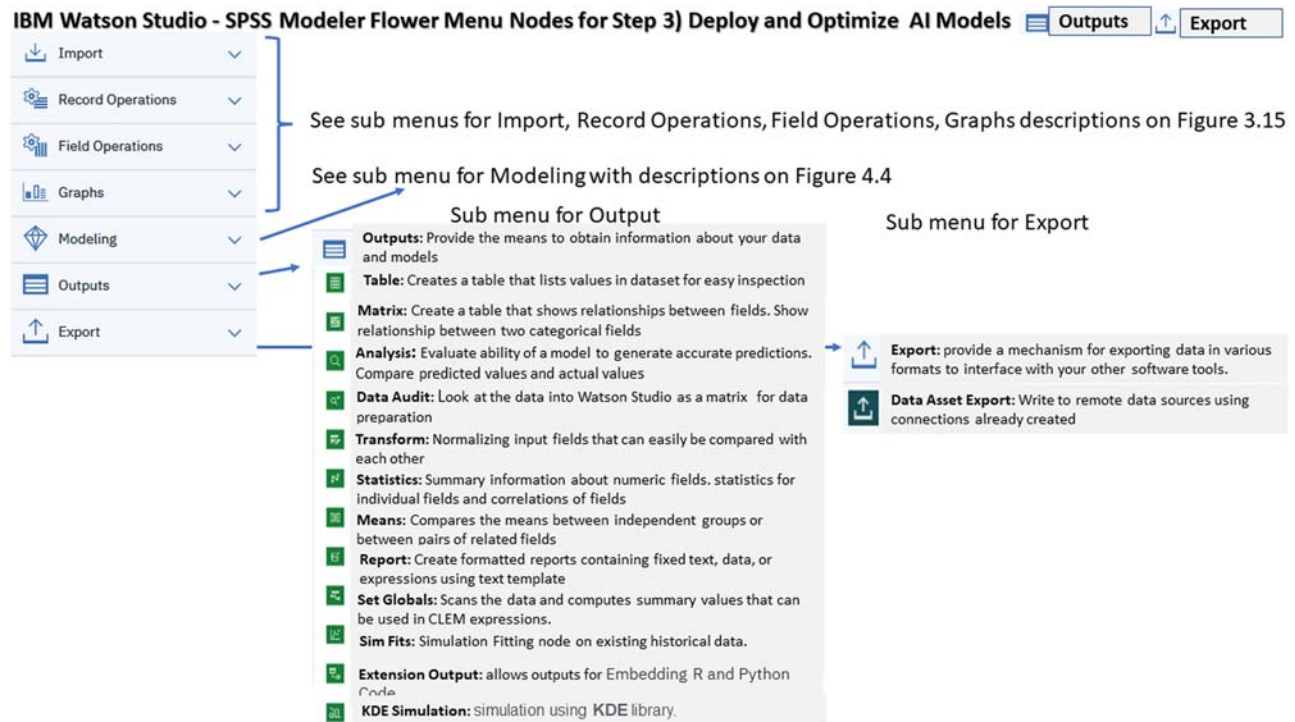


FIGURE 4.5 “IBM Watson SPSS” for submenu: “Output” and “Export.”

- “Analysis”: Evaluates ability of a model to generate accurate predictions. Compares predicted values and actual values.
- “Data Audit”: Looks at the data in Watson Studio as a matrix for data preparation.
- “Transform”: Normalizing input fields that can easily be compared with each other.
- “Statistics”: Summary information about numeric fields. Statistics for individual fields and correlations of fields.
- “Means”: Compares the means between independent groups or between pairs of related fields.
- “Report”: Creates formatted reports containing fixed text, data, or expressions using text templates.
- “Set Globals”: Scans the data and computes summary values that can be used in CLEM expressions.
- “Sim Fits”: Simulation fitting node on existing historical data.
- “Extension Output”: Allows outputs for embedding “R and Python” code.
- “KDE Simulation”: Simulation using KDE using the Ball Tree or KD Tree algorithms for efficient queries.

SPSS Modeler flows > Export

- “Export”: Provides a mechanism for exporting data in various formats to interface with your other software tools.

- “Data Asset Export”: Writes to remote data sources using connections already created.

Note: The best way to learn “IBM Watson SPSS Modeler Flow” is with research tutorials examples.

4.12.2.1 Research 4.1 Tutorial IBM Watson SPSS Modeler Flow for “ML Model for Diabetes”

4.12.2.1.1 Case for research

“Obtain an ML model using Unsupervised Learning with a Clustering algorithm of a diabetes dataset.”

4.12.2.1.2 General objective

“Use IBM Watson Studio IBM SPSS Modeler Flow” to obtain an ML model using Unsupervised Learning with a Clustering algorithm diabetes for a dataset for tests_negative and tested_positive related by “insu” versus “mass and press.”

Note*: This is a continuation of the research 3.5 IBM Watson using SPSS Modeler Flow for “diabetes analysis,” explained in Section 3.4.2.

4.12.2.1.3 Specific objectives

1. Use the “K-Means” algorithm from “IBM Watson using SPSS Modeler” with partitioned data, in two clusters dataset for “tests_negative” and “tested_positive,” related

by “*insu*” versus “*mass and press*” with optimization in memory.

2. Obtain a “*K-Means ML Model*” verifying:
 - a. “*Model Information*”
 - b. “*Cluster Sizes*”
 - c. “*Build Settings*”
 - d. “*Training Summary*”
 - e. “*Clusters exportation*”
 - f. “*Draw Conclusions about K-Means clustering on a diabetes dataset.*”

4.12.2.1.4 Dataset

The dataset “*diabetes.csv**” is based on information at the “*National Institute of Diabetes and Digestive and Kidney Diseases with title: Pima Indians Diabetes Database.*” It has: “768 records,” with “eight fields (attributes)” as indicated in Table 3.18 [19].

Note*: This dataset is available in the companion directory of the book, in the following directory: “. . . \Exercises_book_ABME\CH4\SPSS_MODELER \SPSS_Clusters”.

4.12.2.1.5 Background for “Diabetes”

“*Diabetes*” is a disease that occurs when your blood glucose is too high. Blood glucose is our main source of energy and comes from the food we eat. “*Insulin*,” a hormone made by the pancreas, helps glucose from food get into your cells to be used for energy. There are three types of “*diabetes*”: “*type 1 diabetes*,” “*type 2 diabetes*,” and “*gestational diabetes*” [20]:

- “*Type 1 diabetes*” is when the human body does not produce “*insulin*.”
- “*Type 2 diabetes*” is the most common form of diabetes and it means that the body does not use “*insulin*” properly.
- “*Gestational diabetes*” occurs during pregnancy presenting “*insulin resistance*,” it happens when the “hormones in the placenta” block the mother’s insulin to her body; there is little actual knowledge about the causes of gestational diabetes.

Risk of diabetes can increase for the following factors:

- “*Type 1 diabetes*”: family history, diseases of the pancreas, infection or illness, and other factors.
- “*Type 2 diabetes*”: obesity or being overweight, impaired glucose tolerance*, insulin resistance, ethnic background, gestational diabetes, sedentary lifestyle, family history, age, and other factors. (Note*: *Prediabetes* is a milder form of this condition.)

- “*Gestational diabetes*”: obesity or being overweight, glucose intolerance, family history, age, ethnic background, and other factors.

The “*diabetes*” clinical laboratory tests frequently used are:

- “*Glycated hemoglobin (A1C) test*” indicates the average blood sugar level for the past 2–3 months. It measures the percentage of blood sugar attached to hemoglobin, the oxygen-carrying protein in red blood cells. An “*A1C level of 6.5% or higher*” on two separate tests indicates that you have “*diabetes*.” An “*A1C between 5.7% and 6.4%*” indicates “*prediabetes*.” “*A1C below 5.7%* is considered normal.”
- “*Random blood sugar test*”: a blood sample will be taken at a random time. Regardless of when the person last ate, a random “*blood sugar level of 200 mg/dL equivalent to 11.1 mmol/L—or higher suggests diabetes*.”
- “*Fasting blood sugar test*” is a blood sample taken after an overnight fast. A fasting blood sugar level less than 100 mg/dL (5.6 mmol/L) is normal. A “*fasting blood sugar level from 100 to 125 mg/dL (5.6–6.9 mmol/L) is considered prediabetes*.” If this test is “*126 mg/dL (7 mmol/L) or higher on two separate tests, you have diabetes*.”
- “*Oral glucose tolerance test*” is a test taken after fasting overnight, and the “*fasting blood sugar level*” is measured. Then after a drink of a sugary liquid, blood sugar levels are tested periodically for the next 2 hours.

Note: If “*type 1 diabetes*” is suspected, a “*urine test*” is required to look for the presence of a by-product produced when muscle and fat tissue are used for energy because the body does not have enough insulin to use the available glucose “*ketones*.”

4.12.2.1.6 Procedure

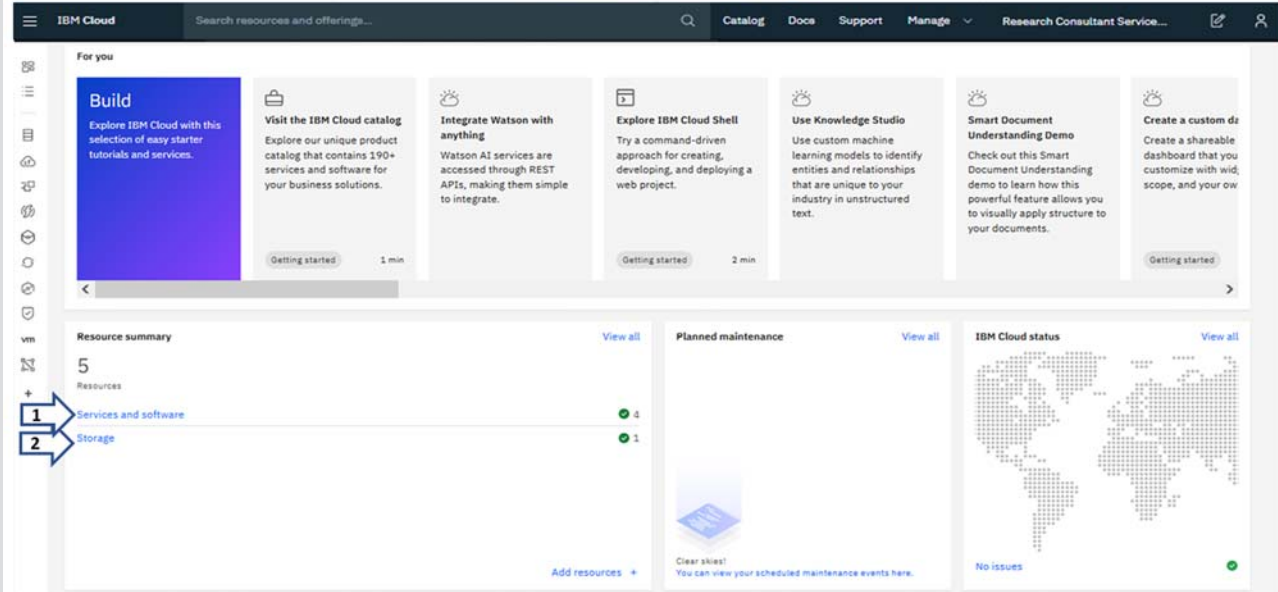
The steps to obtain an *ML model using Unsupervised Learning with a Clustering algorithm for diabetes* applying “*IBM SPSS Modeler Flow*” are summarized in Table of slides 4.1, and each step of the example is visually explained using screen sequences with instructions in figures.

Note: The IBM Cloud website is evolving every day, some screens could be updated. I recommend understanding very well the objectives explained, applying them accordingly with the new screen’s formats and the current IBM Cloud website contents. This research uses the IBM Cloud Pak for Data which is a fully integrated data and AI Watson platform

Table of slides 4.1 steps to obtain an ML model using Unsupervised Learning with a Clustering algorithm for diabetes applying “IBM SPSS Modeler Flow.”

Slide 1 Description
 Login to your IBM Cloud Account at the website: <https://cloud.ibm.com/login> entering your assigned IBMid and Password
 Note: You must have the 4 services available and 1 storage created as per the Chapter 3 tutorial, then click the “Services”

1. Login your account IBM Cloud Account at website: <https://cloud.ibm.com/login> entering your assigned IBMid and Password



You must have the 4 services available, and 1 storage created as per the Chapter 3 tutorial, then click the “Services”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide **Description**
2 In the IBM Cloud—Resource list
Click to expand “Services (4)” and “Storage(1)” and click the service “Watson Studio” to go to the service screen

Screen figure

2. In the IBM Cloud – Resource list

Name	Group	Location	Offering	Status	Tags
> Devices (0)					
> VPC infrastructure (0)					
> Clusters (0)					
> Cloud Foundry apps (0)					
> Cloud Foundry services (0)					
1 ↓ Services (4)					
Natural Language Understanding-ch2	Default	Dallas	Natural Language Understan...	Provisioned	rip
Speech to Text-ch2	Default	Dallas	Speech to Text	Provisioned	rip
Text to Speech-ch2	Default	Dallas	Text to Speech	Provisioned	rip
3 Watson Studio-co	Default	Dallas	Watson Studio	Provisioned	—
2 ↓ Storage (1)					
cloud-object-storage-yp	Default	Global	Cloud Object Storage	Provisioned	—
> Network (0)					
> Cloud Foundry enterprise environments (0)					

Click to expand “Services(4)” and “Storage(1)” and click the service “Watson Studio” to go to the service screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

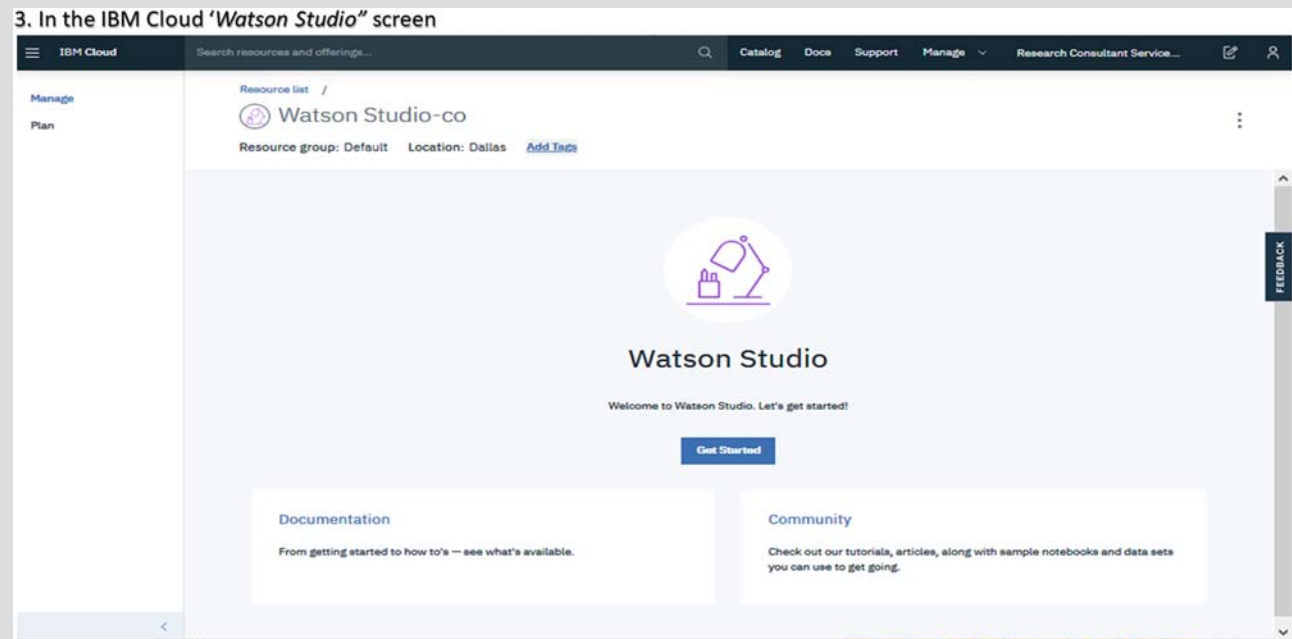
(Continued)

(Continued)

Slide Description

3 In the IBM Cloud "Watson Studio" screen
Note: Press the button "Get Started"

Screen figure



Press the button "Get Started"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 4
Description
In the IBM Watson Studio welcome screen
Click the recently updated project "AI and Cognitive Models"

Screen figure

4. In the IBM Watson Studio welcome screen

Click the recently updated project "AI and Cognitive Models"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

5 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models*
Click on the “*Modeler flows*” already created: “*Diabetes_SPSS*”

5. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models*

The screenshot shows the IBM Watson Studio interface. At the top, there's a navigation bar with 'IBM Watson Studio' and 'Upgrade' button. Below it, the breadcrumb is 'Projects / AI and Cognitive Models'. The main area is divided into tabs: Overview, Assets (selected), Environments, Jobs, Access control, and Settings. A search bar asks 'What assets are you looking for?'. Under 'Data assets', it says '0 assets selected.' and shows a table with one row: 'diabetes.csv' (Data Asset, created by Research Consultant Services, last modified Oct 08, 2021, 12:40 PM). Under 'Modeler flows', it says 'New Modeler flow +' and shows a table with one row: 'Diabetes_SPSS_1' (SPSS Modeler, created by Research Consultant Services, last modified Oct 08, 2021, 01:47 PM). A blue arrow points to the 'Diabetes_SPSS_1' row. On the right, there's a 'Data' panel with 'Load', 'Files', and 'Catalog' tabs, and a dashed box for file upload.

Name	Type	Created by	Last modified
diabetes.csv	Data Asset	Research Consultant Services	Oct 08, 2021, 12:40 PM

Name	Type	Created by	Last modified
Diabetes_SPSS_1	SPSS Modeler	Research Consultant Services	Oct 08, 2021, 01:47 PM

Click on the “*Modeler flows*” already created: “*Diabetes_SPSS*”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

6 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS*
Extent menu: "Modeling," drag "K-Means" icon to the Modeler Flow, join the node with "Type," right click "K-Means" and select "Open," specify in section "Build Options": activate "Use partitioned data," "Number of clusters = 2," press the button "Save," and with right click on "K-Means" node select "Run" to process the model flow for this node

6. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS*

The screenshot displays the IBM Watson Studio interface for a project named "Diabetes_SPSS_1". The main workspace shows a model flow with nodes: "diabetes..." (input), "Filter", "Type", and "class". A "K-Means" node is connected to the "Type" node. A context menu is open over the "K-Means" node, showing options: "Open", "Disconnect", "Preview", "Profile", "Edit", "Delete", "Cache", "Create supernode", and "Run". The "K-Means" configuration panel on the right shows "Use partitioned data" checked, "Number of clusters" set to 2, and "Cluster label" set to String. The "Save" button is highlighted.

Extent menu: "Modeling", drag "K-Means" icon to the Modeler Flow, join the node with "Type", right click "K-Means" and select "Open", specify in section "Build Options": activate "Use partitioned data", "Number of clusters= 2", press the button "Save", and with right click on "K-Means" node select "Run" to process the model flow for this node

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

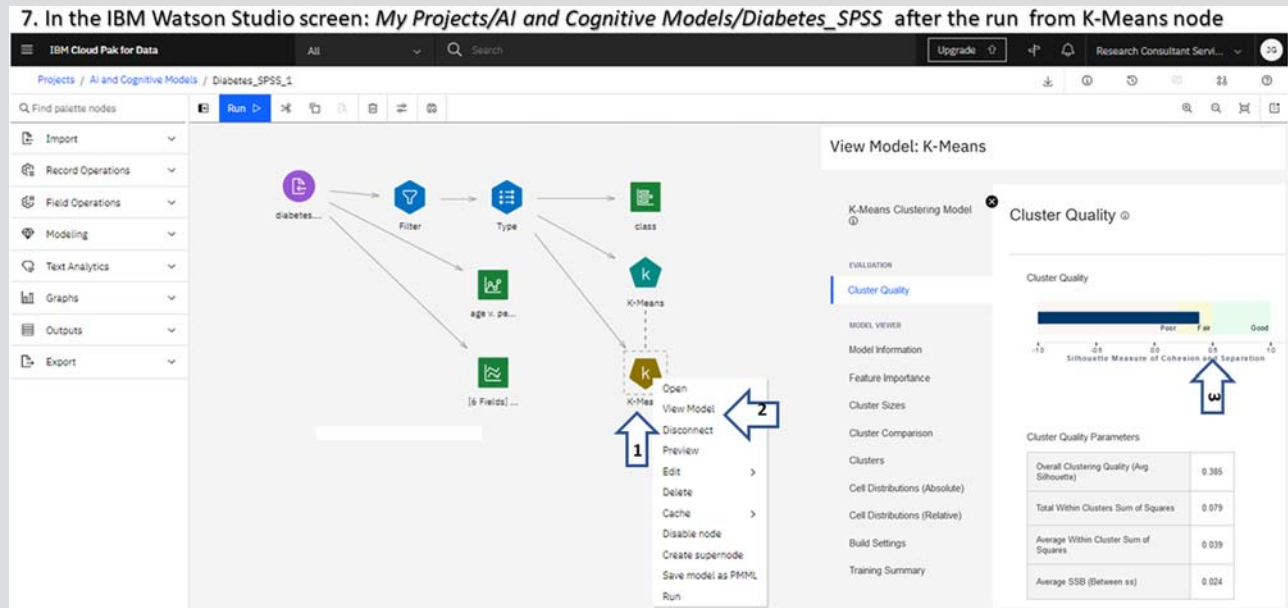
(Continued)

(Continued)

Slide Description

Screen figure

7 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS* after the run from K-Means node
A colored connected icon “K-Means” appears, select and make a right click to see the menu and select “View Model”. It shows a Cluster Quality near to Fair (.5). Where greater than .5 average indicates a reasonable partition of data and less than .2 little evidence.



A colored connected icon “K-Means” appears, select and make right click to see menu and select “View Model”. It shows a Cluster Quality near to Fair (.5). Where greater than .5 average indicates a reasonable partition of data and less than .2 little evidence.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

8 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS/K-Means*: “*Model Info*” & “*Cluster Sizes*”
Select “*Model Information*” observing features = 1, instances cluster 1 = 515, and instances cluster 2 = 253
Select “*Cluster Sizes*” that shows the 2 clusters percentages in a pie chart.
Select “*Diabetes_SPSS*” to go back to the modeler flow

8. IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS/K-Means*: “*Model Info*” & “*Cluster Sizes*”

Select “*Model Information*” observing features=1, instances cluster 1=515, and instances cluster 2=253.
Select “*Cluster Sizes*” that shows the 2 clusters distributions in a pie chart . Select “*Diabetes_SPSS*” to go back to the modeler flow.

Number of instances in each cluster	
Cluster 1	515 (67.06%)
Cluster 2	253 (32.94%)

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

9 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS/K-Means* > Feature: “*Build Settings*” & “*Training Summary*”
Then in IBM Watson Select “*Build Settings*” to check all the settings, and select “*Training summary*” to check the elapsed time for training almost <1 seconds

9. IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS/K-Means*: “*Build Settings*” & “*Training Summary*”

The figure displays two screenshots of the IBM Watson Studio interface. The left screenshot shows the 'Build Settings' screen for a K-Means Clustering Model. The right sidebar menu has 'Build Settings' highlighted with a blue arrow and the number '1'. The main content area contains a table with the following settings:

Setting	Value
Use partitioned data	false
Calculate raw propensity scores	false
Calculate adjusted propensity scores	false
Number of clusters	2
Generate distance field	false
Cluster label	String
Label prefix	cluster
Optimize	Memory
Mode	Simple

The right screenshot shows the 'Training Summary' screen for the same model. The right sidebar menu has 'Training Summary' highlighted with a blue arrow and the number '2'. The main content area contains a table with the following training details:

Property	Value
Algorithm	K-means
Model type	Clustering
Stream	Diabetes_SPSS_1
User	modeler-user
Date built	Mon Nov 30 23:48:11 UTC 2020
Application	IBM SPSS Modeler 18.2.2
Elapsed time for model build	0 hours, 0 mins, 0 secs

Then in IBM Watson Select “*Build Settings*” to check all the settings, and select “*Training summary*” to check the elapsed time for training almost < 1 sec

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

10 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS* - adding a Filter for "Cluster1"
Extent menu: "Record Operations," drag "Select" icon to the Modeler Flow, join the node with "K-Means" yellow icon, right click "Select" and click on "Open," Specify in title section type "Cluster1," activate mode "Include," click the "Expression Builder" icon, and build the expression: '\$KM-K-Means' = "cluster-1," press the "OK" button then the "Save Button"

10. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Diabetes_SPSS - adding a Filter for "Cluster1"

Extent menu: "Record Operations", drag "Select" icon to the Modeler Flow, join the node with "K-Means" result icon, right click "Select" and click on "Open", Specify in title section type "Cluster1", activate mode "Include", click the "Expression Builder" icon, and build the expression: '\$KM-K-Means'="cluster-1", press the "Ok" button then the "Save Button"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 11 Description In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS*—adding a multiplot for “Cluster1” Extent menu: “*Graphs*,” drag “*Multiplot*” icon to the Modeler Flow, join the node with “*Select Cluster1*,” right click “*Multiplot*” and click on “*Open*,” Specify “*x-field = insu*,” on “*y-fields*” click add columns for *mass*, and *press*.” Select the button “*ok*” and select “*panel (discrete) = class*,” then click on “*save*” with right click on “*Multiplot*” node select “*run*” the model flow for this node and on output select the graph generated that open on other window. To go back to Modeler flow, select “*Diabetes_SPSS*”

Screen figure

11. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS* - adding a multiplot for “Cluster1”

Extent menu: “*Graphs*,” drag “*Multiplot*” icon to the Modeler Flow, join the node with “*Select Cluster1*,” right click “*Multiplot*” and click on “*Open*,” Specify “*x-field= insu*,” on “*y-fields*” click “*add columns for mass*, and *press*”. Select the button “*ok*” and select “*panel (discrete)=class*” , then click on “*save*” with right click on “*Multiplot*” node select “*run*” the model flow for this node and on output select the graph generated that open on other window. To go back select Return to Modeler flow

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

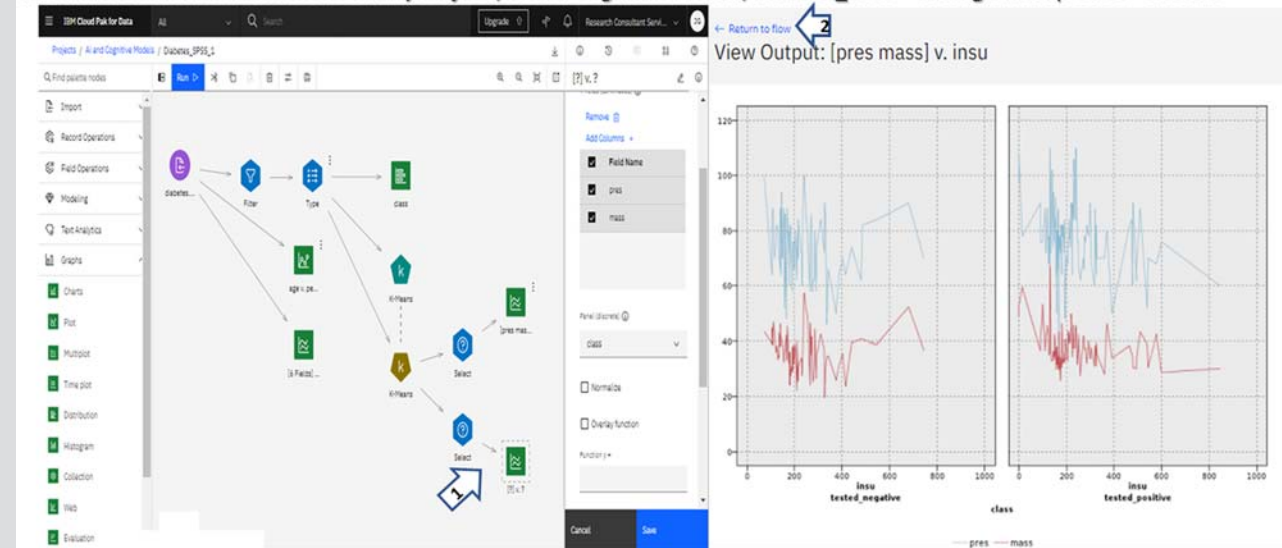
(Continued)

(Continued)

Slide 12 Description In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS*—adding a Web Graph for “Cluster2” Repeat slide 10 instructions to add another: “Record Operations,” with “Select” icon for “Cluster2,” select activate mode “Include,” with “Expression Builder icon,” to build the expression: “\$KM-K-Means’=“cluster-2.” And repeat slide 11 instructions to add another “multiplot” for the same fields to generate a graph for “Cluster2” in another screen

Screen figure

12. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS* - adding a multiplot for “Cluster2”



Repeat slide 10 procedure to add another: “Record Operations”, with “Select” icon for “Cluster2”, select activate mode “Include”, with “Expression Builder icon”, to build the expression: “\$KM-K-Means’=“cluster-2” . And repeat slide 11 to add another “Multiplot” for the same fields to generate a graph for “Cluster2” in another screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

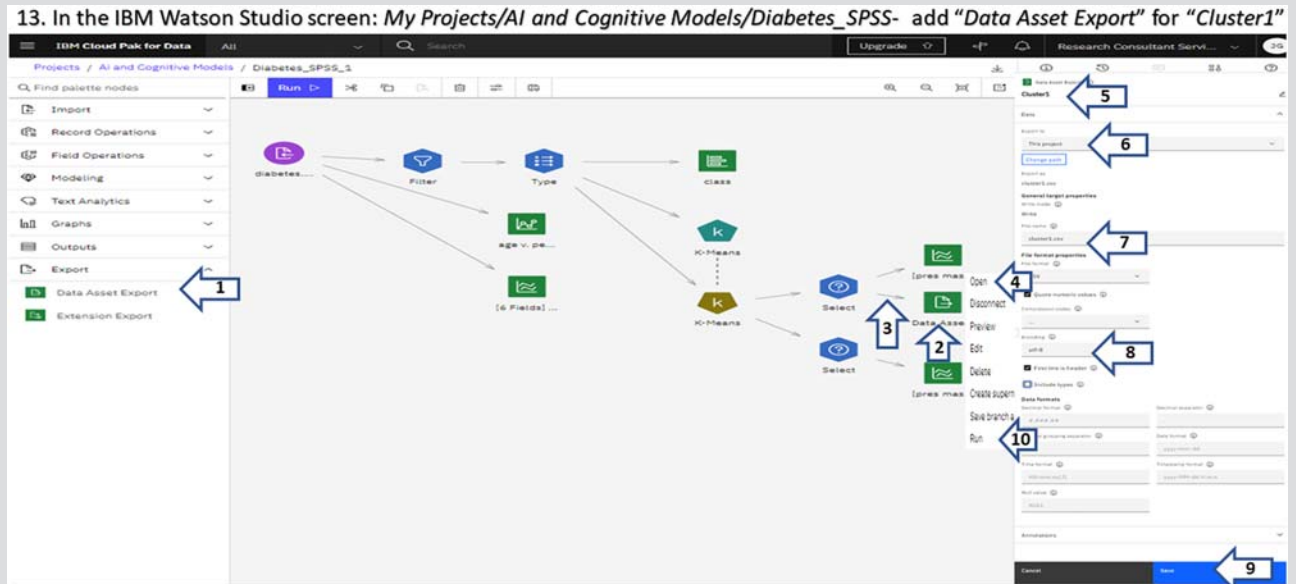
(Continued)

(Continued)

Slide Description

Screen figure

13 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS*—add “Data Asset Export” for “Cluster1” Extent menu: “Export,” drag “Data Asset Export” icon to the Modeler Flow, join the node with “Select Cluster1,” right click “Data Asset Export” and click on “Open,” Specify in title section type “Cluster1,” specify “Export to this project,” filename “cluster1,” and verify the encoding as “utf-8,” press the button “Save.” Right click “Data Asset Export” and select “run” to export



Extent menu: “Export,” drag “Data Asset Export” icon to the Modeler Flow, join the node with “Select Cluster1” , right click “Data Asset Export” and click on “Open”, Specify in title section type “Cluster1”, specify “Export to this project”, filename “cluster1”, and verify the encoding as “utf-8”, press the button “Save”. Right click “Data Asset Export” and select “run” to export

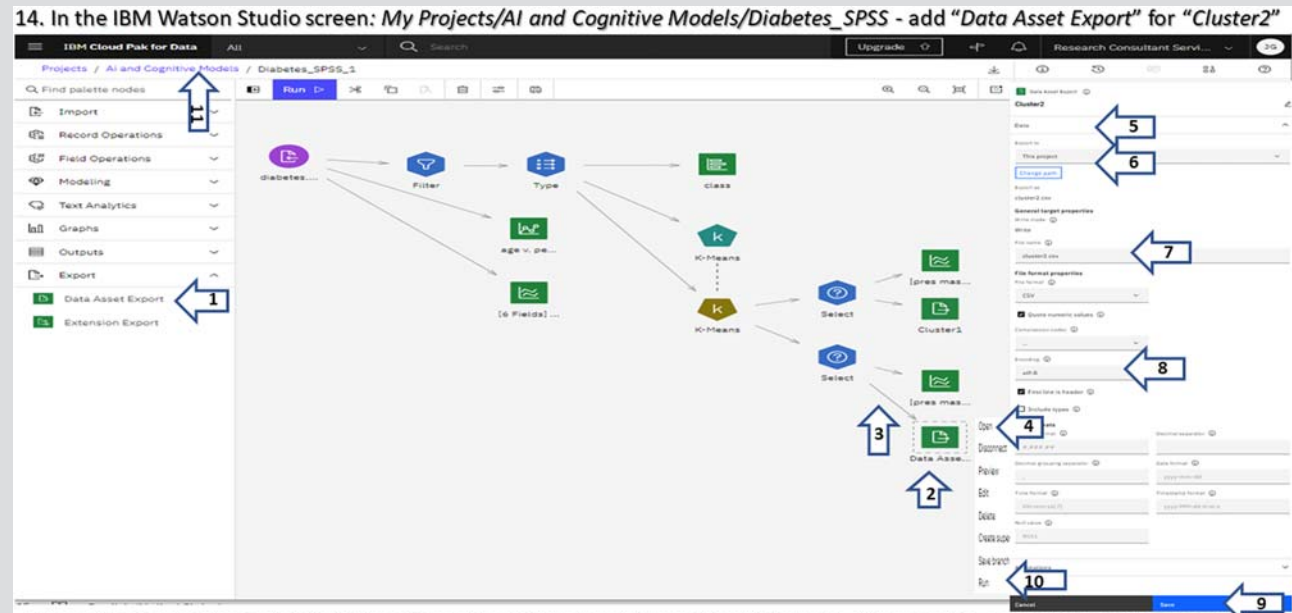
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 14
Description
In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Diabetes_SPSS*—add “Data Asset Export” for “Cluster2”
Repeat the instructions of slide 13 to add another “Data Asset Export” for “Cluster2.” After “run the node” click on the project “AI and Cognitive Models” to go back to the assets screen

Screen figure



Repeat the instructions of slide 13 to add another “Data Asset Export” for “Cluster2”. After run the node click on the project “AI and Cognitive Models” to go back to the assets screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

15 Verify that now there are 3 Data Assets: "diabetes.csv" with the original data of 768 records, "cluster1" with data with the 268 records for "tested_positive," and "cluster2" with data with 500 records for "tested_negative." Finally click on your initials and make a Logout

15. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models: verify the exportation of the two clusters

Name	Type	Created by	Last modified
csv cluster2.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 01:31 PM
csv cluster1.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 01:29 PM
csv diabetes.csv	Data Asset	Research Consultant Services	Oct 08, 2021, 12:40 PM

Verify that now there are 3 Data Assets: "diabetes.csv" with the original data of 768 records, "cluster1" and "cluster2" with data with 500 records separated by their relation of in two plots: "tests_negative" and "tested_positive" related by "insu" vs "mass and press". Finally click on your initials and make a Logout

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

We can have the following conclusions for obtaining an “ML model” using “Unsupervised Learning” with a “Clustering algorithm for diabetes” applying “IBM SPSS Modeler Flow.”

- “IBM SPSS Modeler Flow” allows an easy modeling flow diagram for different “ML models,” specifying in each icon the build options. In this research tutorial example, “K-MEANS” with two-cluster partitioned data was used in two clusters dataset for “tests_negative” and “tested_positive,” related by “insu” versus “mass and press” with optimization in memory as shown in Table of slides 4.1.
- Viewing the generated “K-MEANS” model:
 - A “fair clustering model with value near .5,” using “Euclidean distant” as shown in table of slide 4-1, slide right side 7.
 - The cluster model information shows two clusters at table 4.1 slide 8:
 - “Cluster 1 with 515 (767.06%)”
 - “Cluster 2 with 253 (32.94%)”
 - “Ratio of scale = 2.036”
 - “Multiplot” for cluster1 at slide 11, shows that the “tests_negative” and “tested_positive,” related by “insu” versus “mass and press” with optimization in memory indicated “higher values for insulin” for both results.
 - “Multiplot” for cluster2 at slide 12, shows that the “tests_negative” and “tested_positive,” related by “insu” versus “mass and press” with optimization in memory indicated “lower values for insulin” for both results.
 - Two datasets, one for each cluster was exported for further analysis using other “AI Models.”

Recommendations

- Update the dataset or create a new dataset that include the following more diabetes attributes as: “urine_test” and “hemoglobin_A1c_test.”
- Larger dataset will help to analyze with more precision the “diabetes.”
- Apply more “Machine Learning” (ML) algorithms for classification and regression* for the dataset that describes a population that is under a high risk of the onset of “diabetes.”

Note*: See more “ML Models for this diabetes dataset in: Research 4.5, section 4.12.2.5 MATLAB: Statistics and Machine Learning Toolbox.”

4.12.2.2 Research tutorial 4.2 IBM Watson SPSS Modeler Flow for “Heart disease ML model and deployment”

4.12.2.2.1 Case for research

“Obtain and deploy its ML model applying Decision Tree algorithm under Supervised Learning from a heart diseases dataset.”

4.12.2.2.2 General objective

“Use IBM Watson Studio IBM SPSS Modeler Flow” to obtain an ML model and deploy a Supervised Learning Decision Tree using two classifiers: Classification and Regression (C&R Tree) and “C5.0” algorithms for a heart disease dataset

4.12.2.2.3 Background for “Heart diseases”

“Heart diseases” or “cardiovascular diseases” are a range of conditions that affect the human heart, they are conditions that involve “narrowed or block blood vessels” that can lead to:

- “Heart attack,” when blood flow to a part of the heart is blocked, usually by a “blood clot.”
- “Angina” as a chest pain symptom.
- “Stroke” when a blood vessel feeding the brain gets clogged or bursts.
- And other “heart disorders” such as:
 - “Coronary artery disease.”
 - “Arrhythmias” as a heart rhythm problem.
 - “Congenital disorders” as a heart defects since newborn babies.

The frequent “risk factors” for “heart disease” are [21]:

- “Age,” which usually increases the risk of damaged and narrowed arteries and weakened or thickened heart muscle.
- “Sex”: men have greater risk of heart disease. However, women’s risk increases after menopause.
- “Family history”: a family history of heart disease increases your risk of coronary artery disease.
- “Smoking”: heart attacks are more common in smokers than in nonsmokers.
- “Poor diet,” especially a diet high in fat, salt, sugar, and cholesterol, can contribute to the development of heart disease.
- “High blood pressure” can result in hardening and thickening of arteries, narrowing the vessels through which blood flows.
- “High blood cholesterol levels” in blood can increase the risk of formation of plaques and atherosclerosis.
- “Diabetes” increases your risk of heart disease.
- “Obesity”: excess weight typically worsens other risk factors.
- “Physical inactivity”: the lack of exercise is associated with many forms of heart disease.
- “Stress”: unrelieved stress may damage arteries and worsen other risk factors for heart disease.

4.12.2.2.4 Specific objectives

Use two classifiers: “C&R Tree” and “C5.0” to obtain models for the “Heart_disease.csv” dataset.

TABLE 4.1 Dataset *Heart_disease.csv* fields and descriptions.

Field	Description* 303 instances of patients. Number of fields = 14
age	Age in years (decimal)
sex	Sex = [1: 'male', 0: 'female'] (Boolean)
cp	Chest pain type = [1: 'typical angina'; 2: 'atypical angina'; 3: 'non-anginal pain'] (nominal)
testbps	Resting blood pressure (in mm Hg on admission to the hospital) (decimal)
chol	Serum cholesterol in mg/dL (decimal)
fbc	Fasting blood sugar >120 mg/dL fbc = [1: 'true', 0 = 'false'] (Boolean)
restecg	Resting electrocardiographic results [0: 'normal', 1: 'having ST-T wave abnormality', 2: 'probable or definite left ventricular hypertrophy'] (nominal)
thalach	Maximum heart rate achieved (decimal)
exang	Exercise induced angina exang = [1: 'true', 0: 'false'] (Boolean)
oldpeak	ST depression induced by exercise relative to rest (decimal)
slope	Slope of peak exercise ST segment [1: 'upsloping', 2: 'flat', 3: 'downsloping']
ca	Number of major vessels (0–3) colored by flourosopy. ca = [0,1,2,3] (nominal)
thal	Thal = [3: 'normal', 6: 'fixed defect', 7: 'reversible defect'] (nominal)
num	Predicted value, diagnosis of heart disease; integer valued from 0 (no presence) to 4 (nominal)

Note: This dataset is available in the companion directory of the book, in the following directory: ". . . \Exercises_book_ABME\CH4\SPSS_MODELER\SPSS_Classifier\Heart_disease.csv".

1. Obtain a “ML classifier model using C&R Tree algorithm”
 - a. “Analysis for the C&R Tree” using “Coincide Matrices” and “Performance Evaluation”
 - b. “Create a table for the results from C&R Tree”
2. Obtain a “ML classifier model using C5.0 algorithm”
 - a. “Analysis for the C5.0” using “Coincide Matrices” and “Performance Evaluation”
 - b. “Create a table for the results from C5.0”
3. “Draw conclusions as to which model is better”
4. Create a “Machine Learning web service” to be used for the “deployment of the best classifier model” obtained
5. “Deploy the model using IBM Watson API” using “IBM Cloud Shell” this is equivalent to the “Command URL (cURL)” or “Git command tool”, explained in chapter 2, Section 2.8.2.3.

4.12.2.2.5 Dataset

The dataset “*heart_disease.csv*” is based on information from four databases: *Cleveland*, *Hungary*, *Switzerland*, and the *VA Long Beach*, available at the *UCI Machine Learning Repository* [22]. This database contains 76 attributes, but all published experiments refer to using a subset of 14 of them. In particular, the *Cleveland* database is the only one that has been used by “ML researchers” to this date. The “goal is a field named “Num,” that refers to

the “presence of heart disease in the patient.” It is an integer valued from “0 (no presence) to 4.” Experiments with the “*Cleveland database*” have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0). It has “303 records,” with “14 fields (attributes)” as indicated in [Table 4.1](#).

4.12.2.2.6 Procedure

The steps to obtain an “ML model using Supervised Learning with a Decision Tree algorithm for a heart diseases dataset, and deploy the model” are summarized in *Table of slides 4.2* and each step of the example is visually explained using screen sequences with instructions in figures or in a pdf file format, located in: “. . . \Exercises_book_ABME\CH4\SPSS_MODELER\SPSS_Classifiers\Tutorial Ch 4–7 IBM SPSS heart_disease.pdf”.

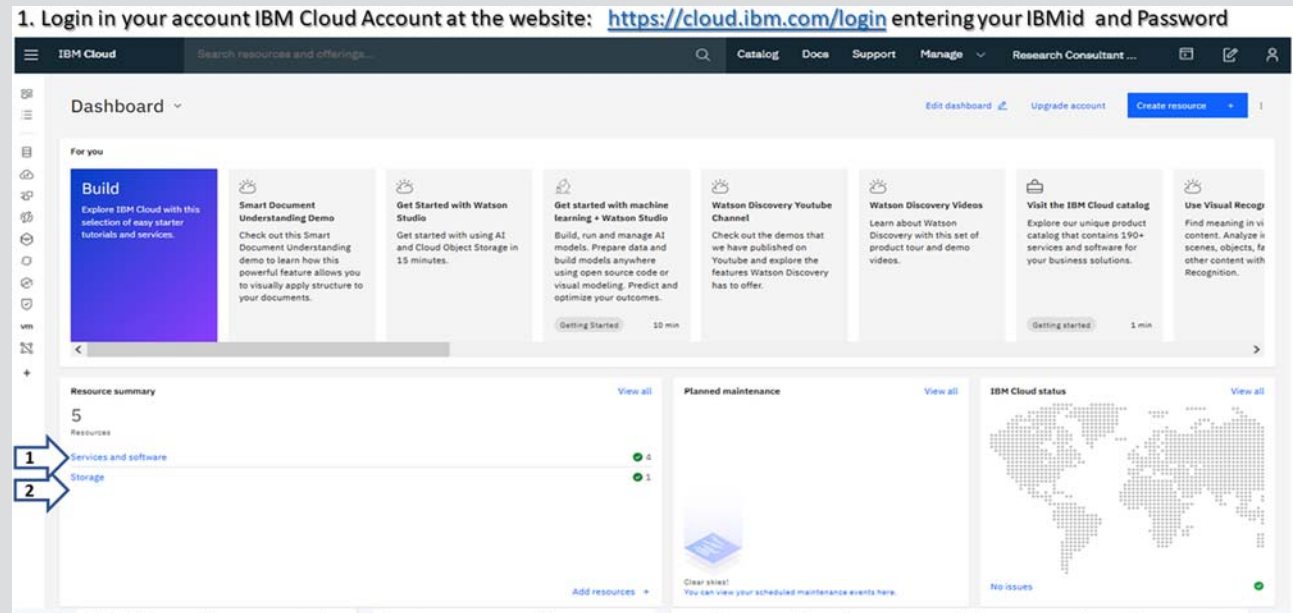
Note: The “IBM Cloud website” is continuously evolving every day by consequence the IBM CLOUD website screens change frequently, some screens could be updated to optimize and simplify the procedures. I recommend understanding very well the objectives, applying them accordingly with the new screen’s formats shown at this table and the current IBM Cloud website contents. This research uses the “IBM Cloud Pak for Data” which is a fully integrated data and AI Watson platform.

Table of slides 4.2 steps to obtain an *“ML model using Supervised Learning with a Decision Tree algorithms for a heart diseases dataset, and deploy the model.”*

Slide Description

Screen figure

1 Login in your IBM Cloud Account at the website: <https://cloud.ibm.com/login> entering your assigned IBMid and Password
 Note: In the *“dashboard”* You must have the 4 services and 1 storage created as per the Chapter 3 tutorial, then click the *“Services”*



In the *“dashboard”* you must have the 4 services and 1 storage created as per the Chapter 3 and first tutorial on Chapter 4, then click the *“Services”*

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

2 In the IBM Cloud—Resource list
Click to expand “Services (4)”
and “Storage(1)” and click the
service “Watson Studio” to go
to the service screen

2. In the IBM Cloud – Resource list

Name	Group	Location	Offering	Status	Tags
Filter by name or IP address... Filter by group or org... Filter... Filter... Filter... Filter...					
Devices (0)					
VPC infrastructure (0)					
Clusters (0)					
Cloud Foundry apps (0)					
Cloud Foundry services (0)					
Services (4)					
Natural Language Understanding-ch2	Default	Dallas	Natural Language Understa...	Provisioned	nlp
Speech to Text-ch2	Default	Dallas	Speech to Text	Provisioned	nlp
Text to Speech-ch2	Default	Dallas	Text to Speech	Provisioned	nlp
Watson Studio-co	Default	Dallas	Watson Studio	Provisioned	—
Storage (1)					
cloud-object-storage-yp	Default	Global	Cloud Object Storage	Provisioned	—
Network (0)					
Cloud Foundry enterprise environments (0)					
Functions namespaces (0)					
Apps (0)					

Click to expand “Services(4)” and “Storage(1)” and click the service “Watson Studio” to go to the service screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

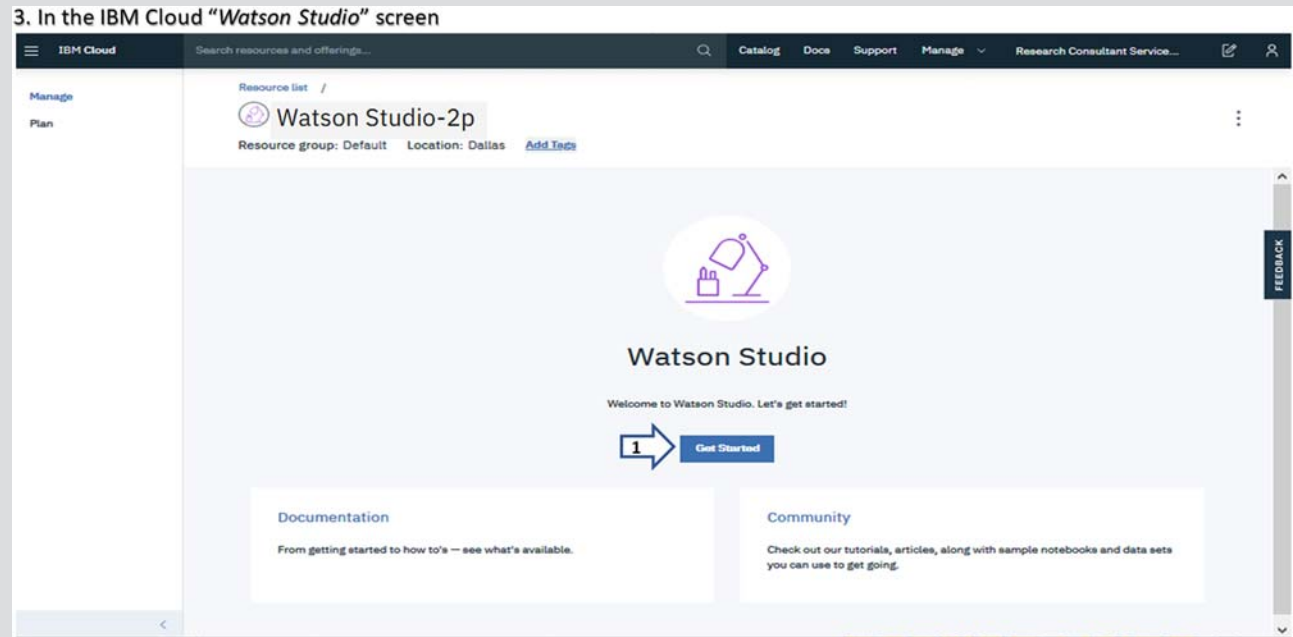
(Continued)

(Continued)

Slide Description

3 In the IBM Cloud "Watson Studio" screen
Note: Press the button "Get Started"

Screen figure



Press the button "Get Started"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

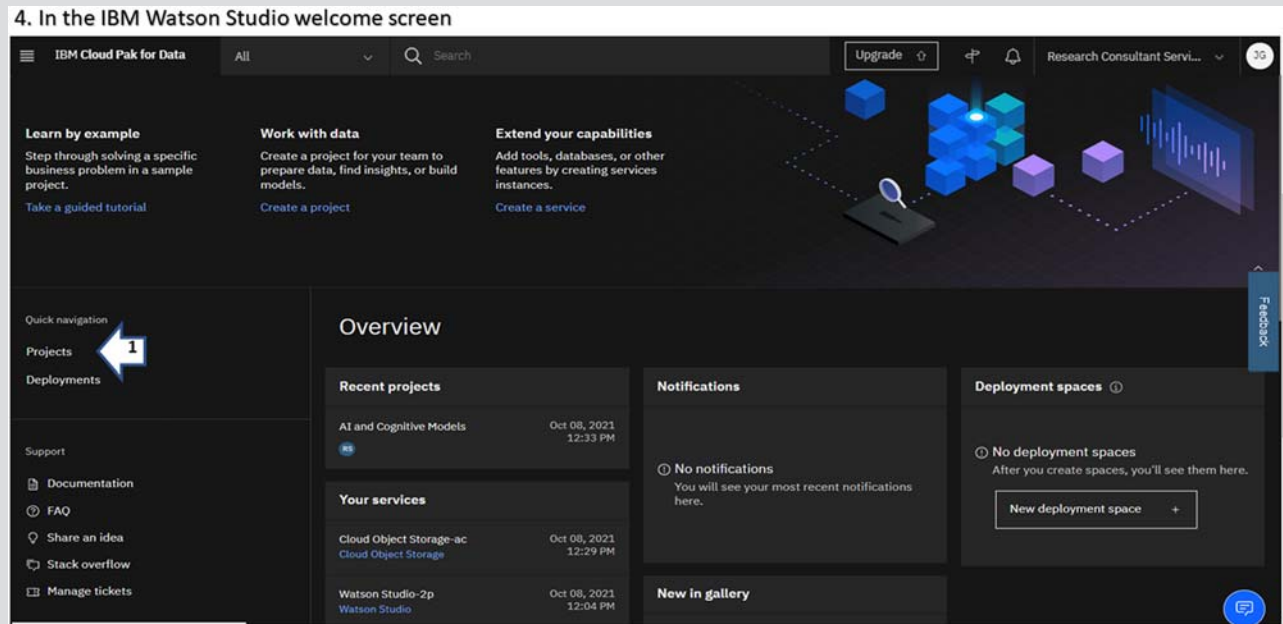
(Continued)

(Continued)

Slide Description

Screen figure

4 In the IBM Watson Studio welcome screen
Click on project as shown in the slide, then select your project "AI and Cognitive Models"



Click on project as shown in the slide, then select your project "AI and Cognitive Models"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

5 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models* in the tab “Assets”
On the project “*AI and Cognitive Models*” slide click on “*New Data asset*” as shown.

5. In the IBM Watson Studio projects screen: “*My Projects/AI and Cognitive Models*” in the tab “Assets”

The screenshot displays the IBM Watson Studio interface for the project "AI and Cognitive Models". The "Assets" tab is active, showing a search bar and a "New Data asset" button with an information icon. Below this, there is a table of existing data assets:

Name	Type	Created by	Last modified
CSV cluster1.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 05:21 PM
CSV cluster2.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 05:21 PM
CSV diabetes.csv	Data Asset	Research Consultant Services	Oct 08, 2021, 12:40 PM

Below the table, there is a section for "Modeler flows" with a "New Modeler flow" button. A table of existing modeler flows is also visible:

Name	Type	Created by	Last modified
Diabetes_SPSS_1	SPSS Modeler	Research Consultant Services	Oct 11, 2021, 05:21 PM

Click on the project “*AI and Cognitive Models*” click on “*New Data asset*” as shown in the slide

(Continued)

(Continued)

Slide Description

Screen figure

6 In the IBM Watson Studio screen for: *My Projects > AI and Cognitive Models* in the tab "Assets"
Load asset using "browse," select the dataset "Heart_disease.csv" in the data companion directory and press the button "Open"

6. In the IBM Watson Studio screen for: *My Projects > AI and Cognitive Models* in the tab "Assets"

Load asset using "browse", select the data set "Heart_disease.csv" in the data companion directory and press the button "Open"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

7 IBM Watson Studio “My Projects/AI and Cognitive Models” section Assets: observe that asset “Heart_disease.csv” is available
Click on the button “Add to project”

7. IBM Watson Studio “My Projects / AI and Cognitive Models” section Assets: observe that asset “Heart_disease.csv” is available

The screenshot shows the IBM Watson Studio interface. At the top, there is a navigation bar with 'IBM Cloud Pak for Data' and 'All' filters. Below that, the 'My Projects / AI and Cognitive Models' section is active. The 'Assets' tab is selected, showing a search bar and a table of data assets. The table has columns for Name, Type, Created by, and Last modified. The first row is 'Heart_disease.csv', which is highlighted with a red arrow labeled '1'. Other rows include 'cluster1.csv', 'cluster2.csv', and 'diabetes.csv'. To the right of the table, there is a 'Load' button and a 'Files' section with a 'Drop files here or browse for files to upload.' prompt. A red arrow labeled '2' points to the 'Add to project' button in the top right corner.

Name	Type	Created by	Last modified
CSV Heart_disease.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 05:43 PM
CSV cluster1.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 05:21 PM
CSV cluster2.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 05:21 PM
CSV diabetes.csv	Data Asset	Research Consultant Services	Oct 08, 2021, 12:40 PM

Click on the button “Add to project”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

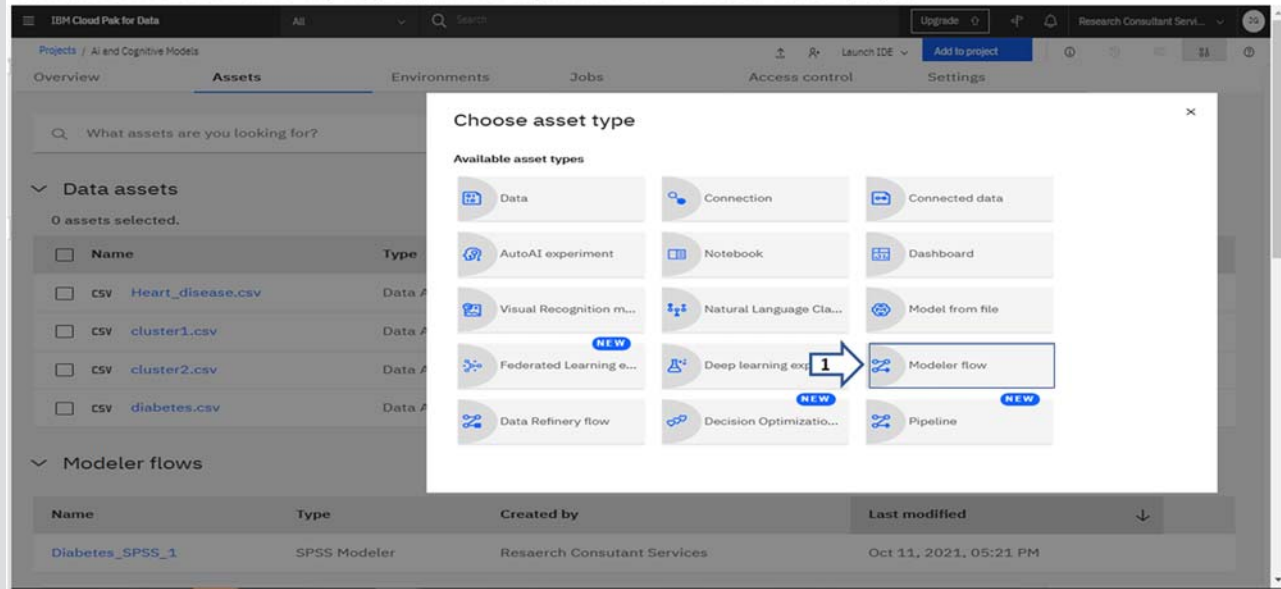
(Continued)

Slide Description

Screen figure

8 In the IBM Watson Studio "My Projects/AI and Cognitive Models" section "Add to project"
Click on the button "Modeler flow"

8. In the IBM Watson Studio "My Projects / AI and Cognitive Models" section "Add to project"



Click on the button "Modeler flow"

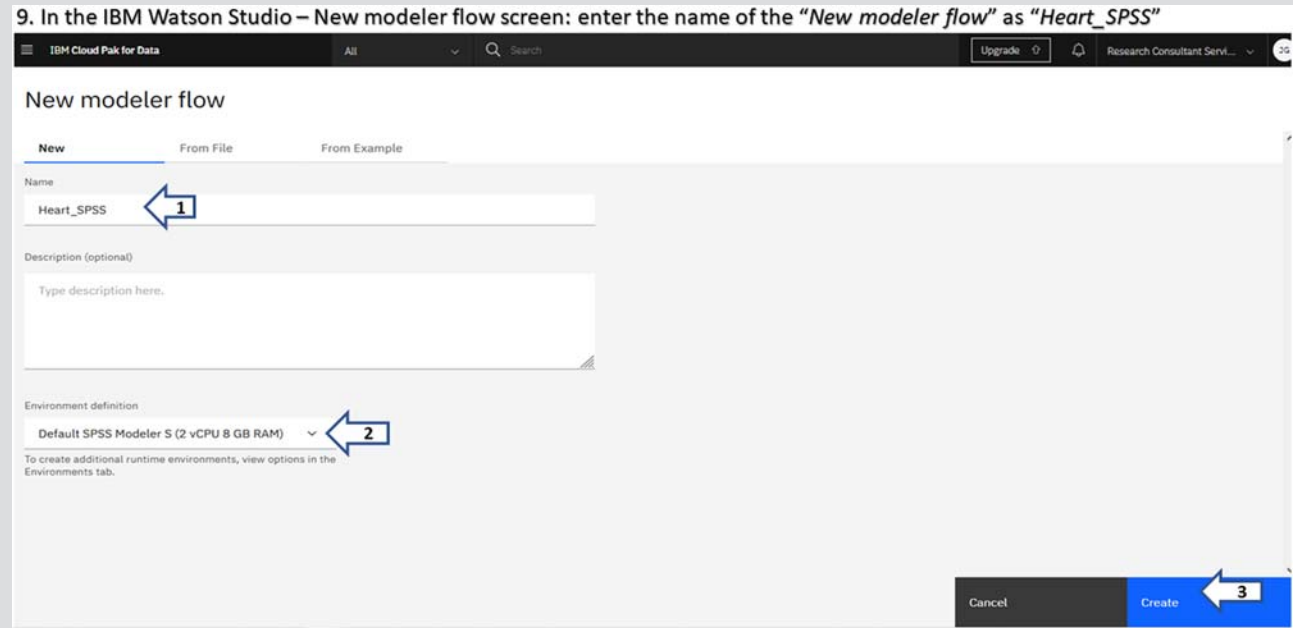
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 9 Description
In the IBM Watson Studio—
New modeler flow screen:
enter the name of the “New
modeler flow” as
“Heart_SPSS”
Select the options:
“Environment definition” as
“Default SPSS Modeler S
(2 vCPU 8 GRAM)” and click
on the button “Create”

Screen figure



Select the options: “Environment definition” as “Default SPSS Modeler S (2 vCPU 8 GRAM)” and click on the button “Create”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

10 In the IBM Watson Studio "My Projects/AI and Cognitive Models/Heart_SPSS" screen: "Data Asset"
Select "Import" to expand its menu, click and drag "Data Asset" to create the first node of the model flow

10. In the IBM Watson Studio My Projects/AI and Cognitive Models/Heart_SPSS screen: "Data Asset"

The screenshot shows the IBM Watson Studio interface. The top navigation bar includes 'IBM Cloud Pak for Data', 'All', a search bar, and an 'Upgrade' button. The breadcrumb trail is 'Projects / AI and Cognitive Models / Heart_SPSS'. The left sidebar contains a 'Find palette nodes' search bar and a list of categories: 'Import', 'Data Asset', 'User Input', 'Sim Gen', 'Extension Import', 'Record Operations', 'Field Operations', 'Modeling', 'Text Analytics', 'Graphs', 'Outputs', and 'Export'. The 'Import' category is expanded, and the 'Data Asset' node is highlighted with a blue arrow labeled '2'. A blue arrow labeled '1' points to the 'Run' button in the top toolbar. The main canvas is empty with a 'Get started' message: 'Drag a node from the palette or a data file from your computer to your flow canvas.' On the right side, there is a 'Data Assets' section with the text 'You may upload multiple data assets.' and a 'Drop files here or browse for files to upload.' prompt.

Select "Import" to expand its menu, click and drag "Data Asset" to create the first node of the model flow

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

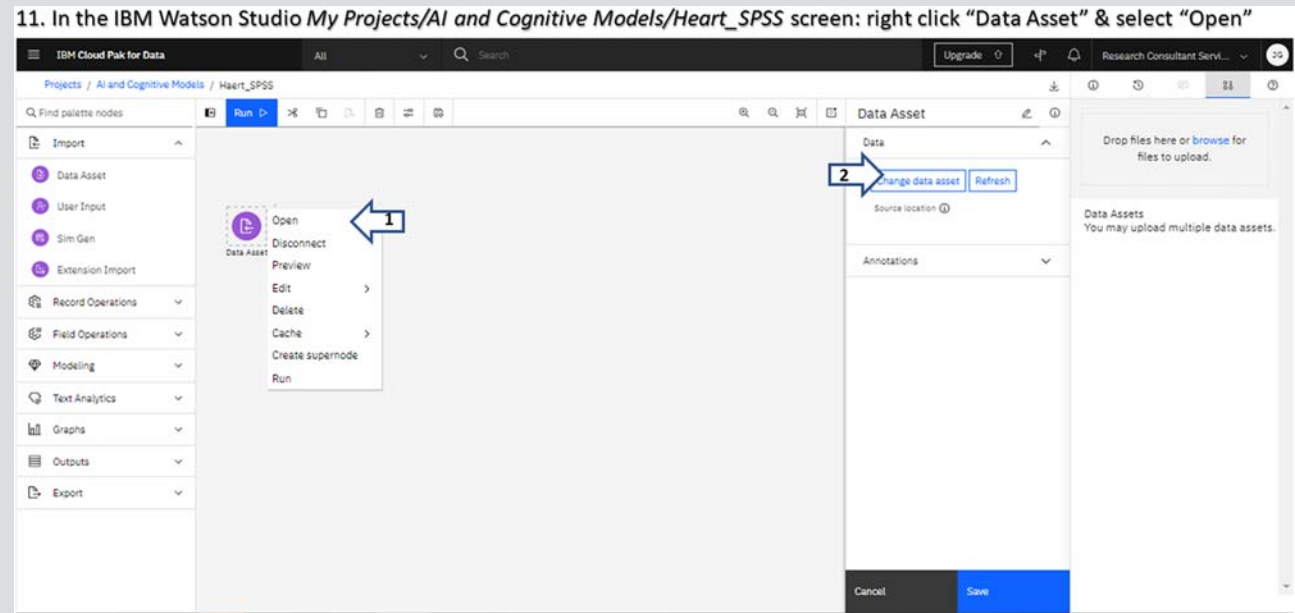
(Continued)

(Continued)

Slide Description

Screen figure

11 In the IBM Watson Studio *My Projects/AI and Cognitive Models/Heart_SPSS* screen: right click “Data Asset” and select “Open”
Click the button “Change data asset”



Click the button “Change data asset”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

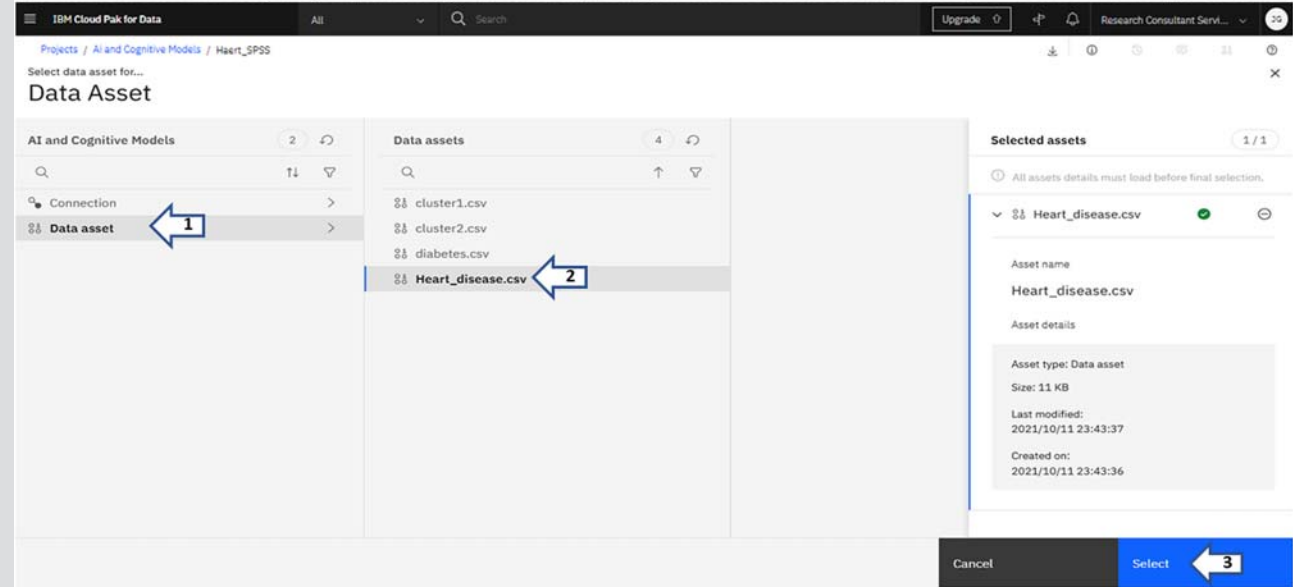
(Continued)

Slide Description

Screen figure

12 IBM Watson Studio My Projects/AI and Cognitive Models/Heart_disease screen: observe Data Assets source location, press "Save"
observe Data Assets source location, press "Save"
Select: "Data assets,"
"Heart_disease.csv" then select the button "Select"

12. IBM Watson Studio My Projects/AI & Cognitive Models/Heart_disease screen: observe Data Assets source location, press "Save"



Select: "Data assets", "Heart_disease.csv" then select the button "Select"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

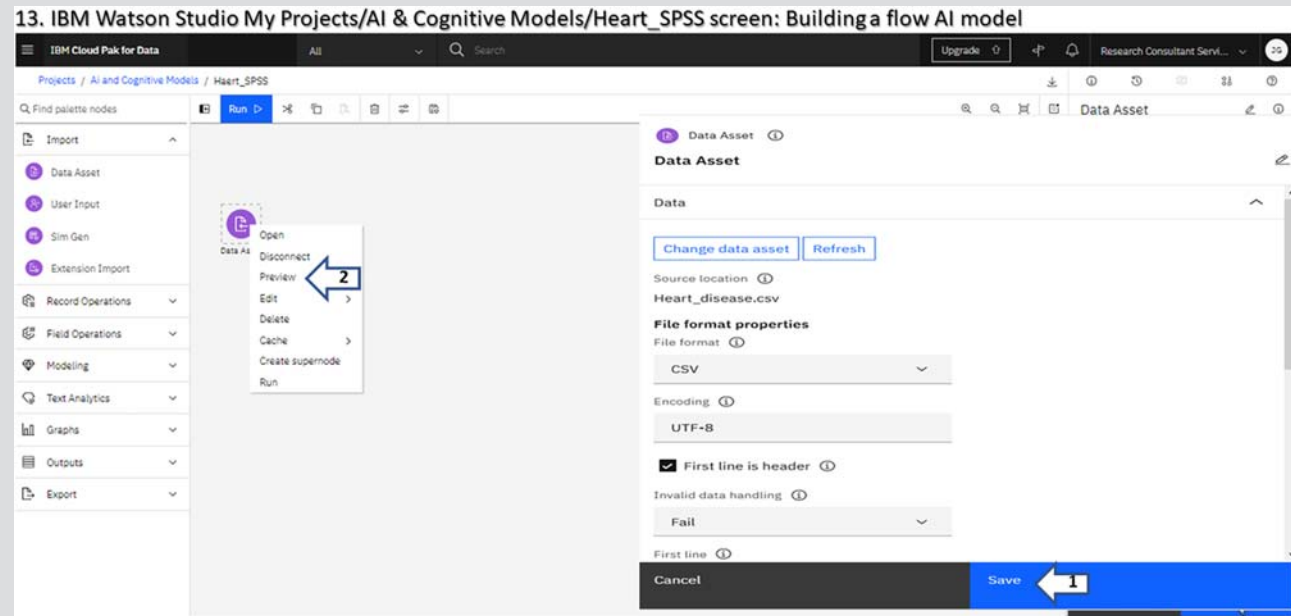
(Continued)

(Continued)

Slide Description

Screen figure

13 IBM Watson Studio My Projects/AI and Cognitive Models/Heart_SPSS screen: Building a flow AI model "Save", make a right click on "Data Asset" and select "Preview"



"Save", make a right click on "Data Asset" and select "Preview"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

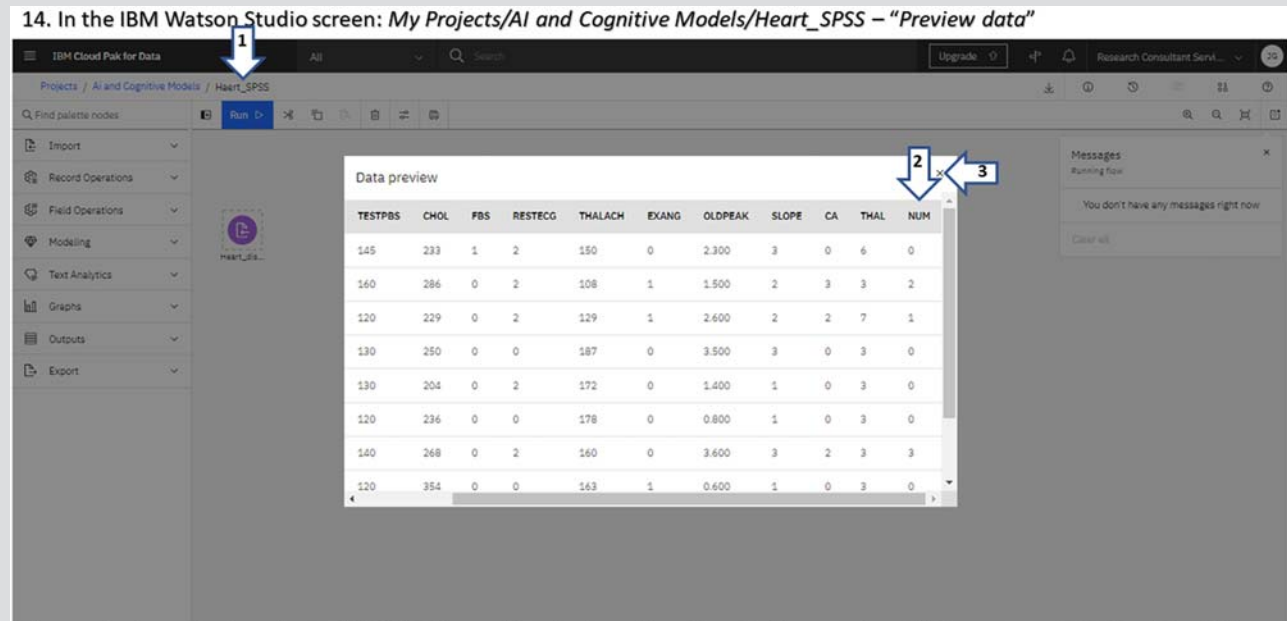
(Continued)

Slide Description

Screen figure

14 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS - "Preview data"*
Observe in data tab, the column "NUM" is the predicted values from "nominal range: 0 to 4." To return to SPSS Modeler click on "close window icon" in the upper right corner

14. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS - "Preview data"*



TESTPBS	CHOL	FBS	RESTECG	THALACH	EXANG	OLDPEAK	SLOPE	CA	THAL	NUM
145	233	1	2	150	0	2.300	3	0	6	0
160	286	0	2	108	1	1.500	2	3	3	2
120	229	0	2	129	1	2.600	2	2	7	1
130	250	0	0	187	0	3.500	3	0	3	0
130	204	0	2	172	0	1.400	1	0	3	0
120	236	0	0	178	0	0.800	1	0	3	0
140	268	0	2	160	0	3.600	3	2	3	3
120	354	0	0	163	1	0.600	1	0	3	0

Observe in data tab, the column "NUM" is the predicted values from "nominal range: 0 to 4". To return to SPSS Modeler click on "close window icon" in the upper right corner

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

15 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS*: Specify the “Target” for the “Classifier”
Select “Field Operations” to expand its menu, click and drag “Type” icon to the Modeler Flow, join the nodes, right click “Type” and select “Open.” In “Type Operations” configure field “NUM” as “Measure as “nominal,” “Role” as “Target” and click “Save”

15. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Heart_SPSS: Specify the “Target” for the “Classifier”

The screenshot shows the IBM Watson Studio interface. On the left, the 'Field Operations' menu is expanded, and the 'Type' icon is being dragged to the Modeler Flow. The 'Type' node is connected to the 'heart_spss' node. A context menu is open over the 'Type' node, with 'Open' selected. On the right, the 'Type Operations' settings are visible. The 'Default Mode' is set to 'Read metadata'. The 'Type Operations' table is shown below:

Field	Measure	Role	Value Mode	Values	Check
# RESTECK	Continuous	Input	Read		None
# THALACH	Continuous	Input	Read		None
# EXANG	Continuous	Input	Read		None
# OLDPEAK	Continuous	Input	Read		None
# SLOPE	Continuous	Input	Read		None
# CA	Categorical	Input	Read		None
# THAL	Categorical	Input	Read		None
# NUM	Nominal	Target	Read		None

Select “Field Operations” to expand its menu, click and drag “Type” icon to the Modeler Flow, join the nodes, right click “Type” and select “Open.” In “Type Operations” configure field “NUM” as “Measure as “nominal,” “Role” as “Target” and click “Save”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

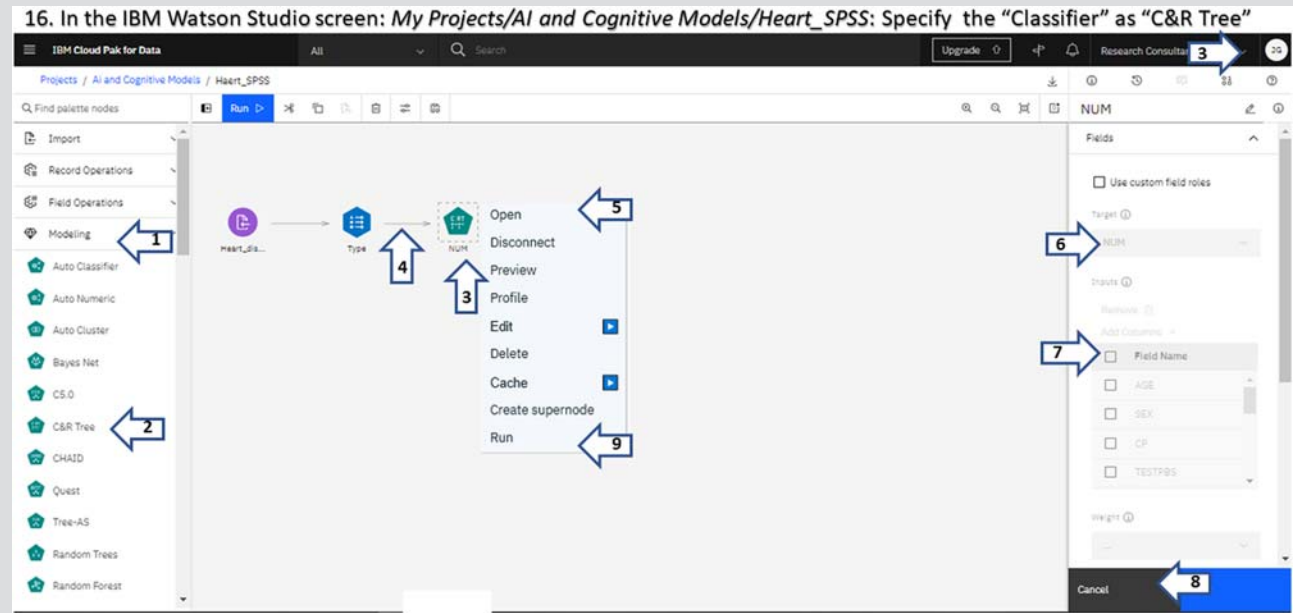
(Continued)

(Continued)

Slide Description

Screen figure

16 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS*: Specify the “Classifier” as “C&R Tree”
Select “Modeling” to expand its menu, click and drag “C&R Tree” icon to the Modeler Flow, join the nodes with “Type,” select “C&R Tree” icon make a right click and select “Open.” Verify the default values of “Target” and “Inputs” and click “Save.” Select again “C&R Tree” icon make a right click and select “Run.”



Select “Modeling” to expand its menu, click and drag “C&R Tree” icon to the Modeler Flow, join the nodes with “Type”, select “C&R Tree” icon make a right click and select “Open”. Verify the default values of “Target” and “Inputs” and click “Save”. Select again “C&R Tree” icon make a right click and select “Run”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

17 In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Heart_SPSS: Add an output "Table"
Select "Output" to expand its menu, click and drag "Table" icon to Modeler Flow, join it with the result of "C&R Tree" in the yellow icon, select "Table" make a right click and select "Run." In the Output Panel double click "Table (16 fields, 303 records)" to see result, observe the classifier results in "\$R-NUM" and "RC-NUM." Return to modeler flow with click in "Close x" to return to the flow

17. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Heart_SPSS: Add an output "Table"

The screenshot shows the IBM Watson Studio interface. On the left, the 'Outputs' menu is expanded, and the 'Table' icon is highlighted. In the center, a modeler flow is visible with a 'Table' node connected to a 'C&R Tree' node. A right-click context menu is open over the 'Table' node, with 'Run' selected. On the right, the 'Outputs' panel shows a table with 16 fields and 303 records. The table data is as follows:

AGE	SEX	CP	TESTFBS	CHOL	FBS	RESTECG	THALACK	EXANG	OLDPEAK	SLOPE	CA	THAL	NUM	SR-NUM	RC-NUM
63	M	1	145	233	1	2	150	0	2.300	3	0	4	0	0	0.530
67	F	4	140	286	0	2	138	1	1.900	2	3	3	2	0	0.760
67	F	4	120	229	0	0	129	1	2.400	2	2	7	1	3	0.429
37	F	3	130	250	0	0	147	0	3.500	3	0	3	0	0	0.760
41	0	2	130	204	0	2	170	0	1.400	1	0	3	0	0	0.760
36	F	2	120	236	0	0	178	0	0.800	1	0	3	0	0	0.760
42	0	4	140	268	0	2	140	0	3.400	3	2	3	3	0	0.760
37	0	4	120	154	0	0	143	1	0.400	1	0	3	0	0	0.760
43	F	4	130	254	0	2	147	0	1.400	2	1	7	2	2	0.441
53	F	4	140	203	1	2	156	1	3.100	3	0	7	1	3	0.441
37	F	4	140	190	0	0	148	0	0.400	2	0	4	0	0	0.434
36	0	2	140	244	0	2	153	0	1.300	2	0	3	0	0	0.760

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

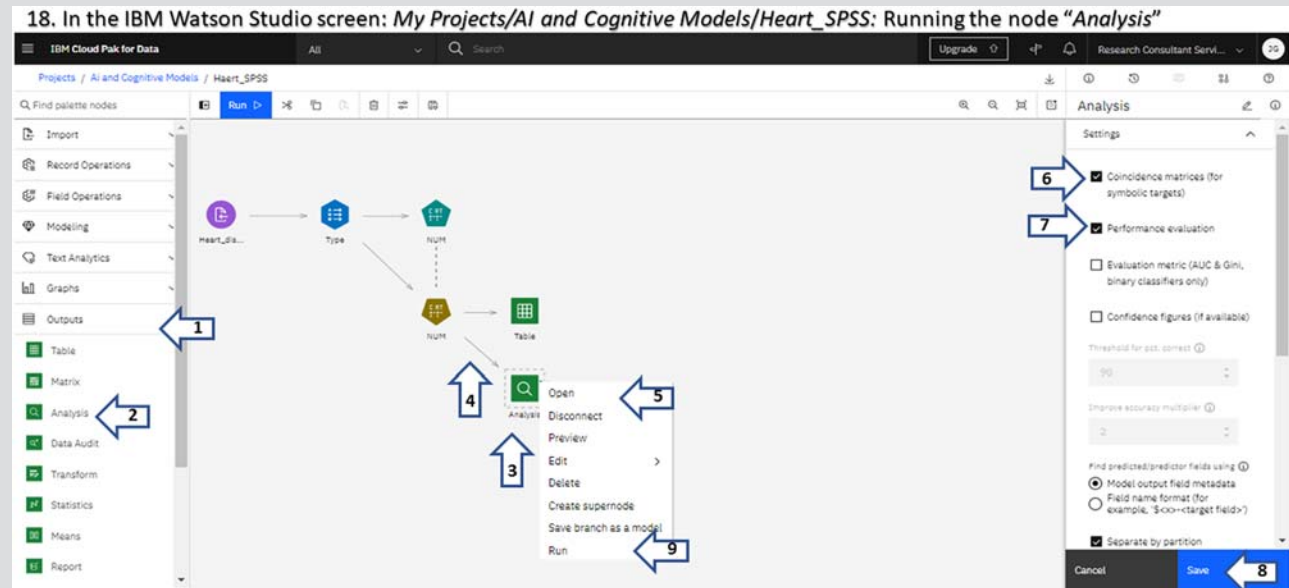
(Continued)

(Continued)

Slide Description

Screen figure

18 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS*: Running the node "Analysis"
Select "Output" to expand its menu, click and drag "Analysis" icon to Modeler Flow, join it with the result of "C&R Tree" in the yellow icon, select "Analysis" make a right click and select "Open."
In the setting panel activate "Coincidence matrices," and "Performance evaluation" click the button "Save." Select again "Analysis" make a right click and select "Run"



Select "Output" to expand its menu, click and drag "Analysis" icon to Modeler Flow, join it with the result of "C&R Tree" in the yellow icon, select "Analysis" make a right click and select "Open". In the setting panel activate "Coincidence matrices," and "Performance evaluation" click the button "Save". Select again "Analysis" make a right click and select "Run".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

19 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS: Results "C&R Tree" Analysis*
In the Output Panel click the symbol in "Analysis" output to see the result in another screen the original "NUM" with the correct classification in "\$R-NUM" of 63.37% for the classifier "C&R Tree," beside the results for "Coincidence Matrix" and "Performance Evaluation"

19. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS: Results "C&R Tree" Analysis*

Results for output field NUM
Comparing \$R-NUM with NUM
Correct 192 63.37%
Wrong 111 36.63%
Total 303

Coincidence Matrix for \$R-NUM (rows show actuals)

	0	2	3
0	157	6	1
1	36	9	10
2	12	18	6
3	10	8	17
4	3	4	6

Performance Evaluation

0	0.286
2	1.214
3	1.303

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

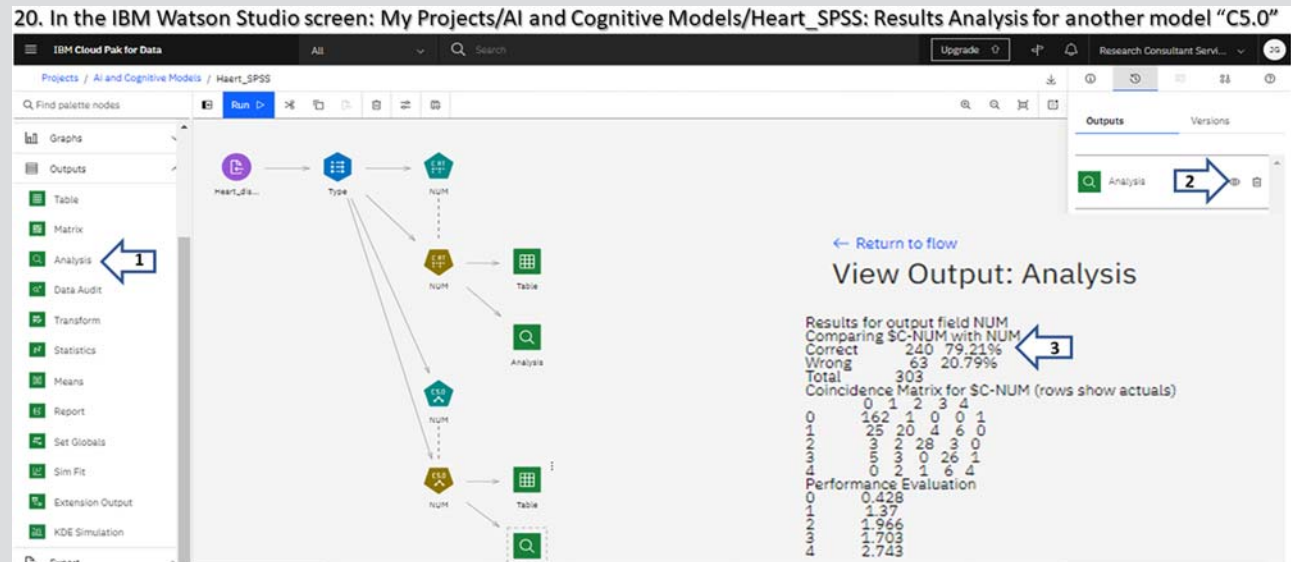
(Continued)

(Continued)

Slide Description

Screen figure

20 In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Heart_SPSS: Results Analysis for another model "C5.0"
Applying the same procedure add: a classifier "C5.0" from "Modeling" a "Table" and "Analysis" as shown. Run the node "Analysis" for the branch of classifier "C5.0". In the Output Panel double click "Analysis" to see the result in another screen the original "NUM" with the correct classification in "\$R-NUM = predicted value of 79.21%" for the classifier "C5.0," beside better results for "Coincidence Matrix" and "Performance Evaluation"



Applying the same procedure add: a classifier "C5.0" from "Modeling", a "Table" and "Analysis" as shown. Run the node "Analysis" for the branch of classifier "C5.0". In the Output Panel double click "Analysis" to see the result in another screen the original "NUM" with the correct classification in "\$R-NUM=predicted value of 79.21%" for the classifier "C5.0", beside better results for "Coincidence Matrix" and "Performance Evaluation"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

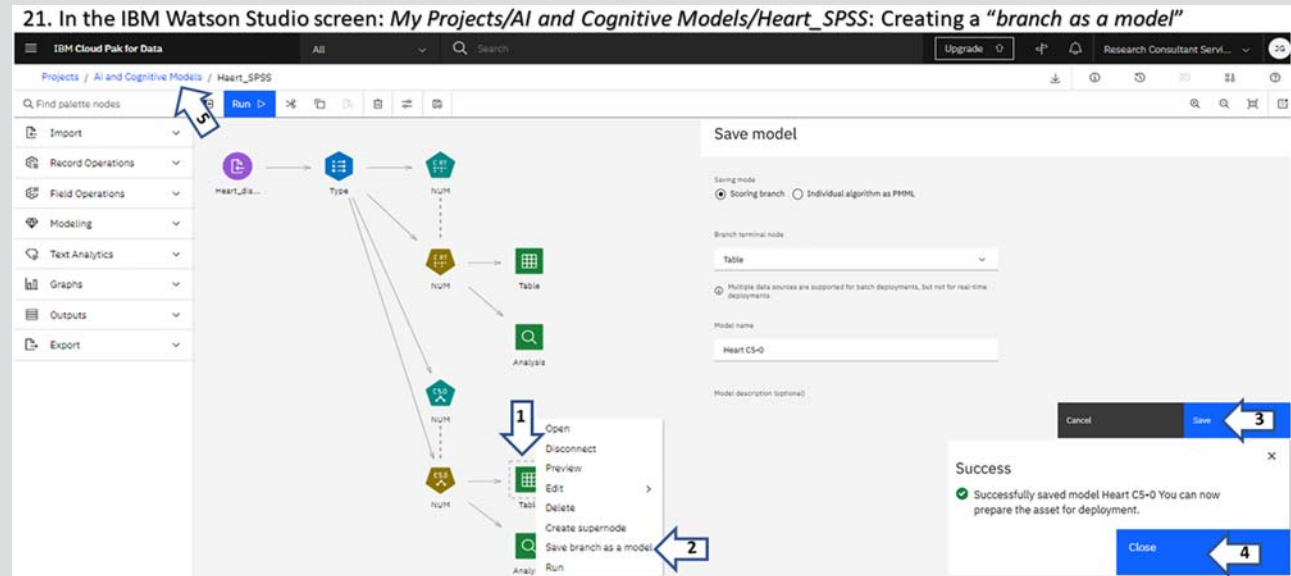
(Continued)

(Continued)

Slide Description

Screen figure

21 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS: Creating a "branch as a model"*
Going back to the modeler flower select the "Table" from the branch of the classifier "5.0," make a right click to open the screen menu and select "Save branch as a model." A windows to save model appears, specify sending as "Table", model name = "Heart 5-0" and click "Save". Then a "Success windows" indicates that you now can prepare the asset for deployment, click "Close"



Going back to the modeler flower select the "Table" from the branch of the classifier "5.0", make a right click to open the screen menu and select "Save branch as a model". A windows to save model appears specify sending as "Table", model name="Heart 5-0" and click "Save". Then a "Success windows" indicates that you now can prepare the asset for deployment,click "Close".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

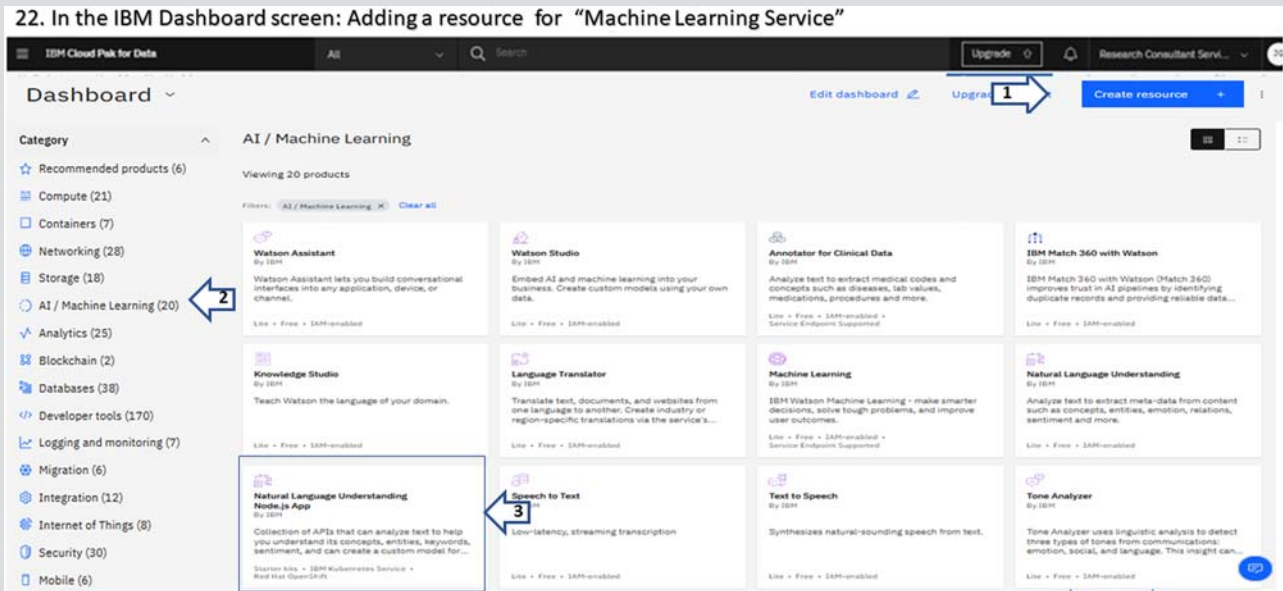
(Continued)

(Continued)

Slide Description

Screen figure

22 In the IBM Dashboard screen: adding a resource for “Machine Learning Service”
At the “IBM Dashboard” screen select “Create Resource”, select filter of “AI/ Machine Learning”, and click on “Machine Learning” Service.



At the “IBM Dashboard” screen select “Create Resource”, select filter of “AI/Machine Learning”, and click on “Machine Learning” service.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

23 In the IBM Dashboard screen: Adding a resource for “Machine Learning Service” At creating the “Machine Learning Service” select closet location, accept “license agreement” and click on “Create”

23. In the IBM Dashboard screen: Adding a resource for “Machine Learning Service”

The screenshot displays the IBM Cloud dashboard for adding a resource. The main content area is divided into two sections: "Select a location" and "Select a pricing plan".

Select a location: A dropdown menu is shown with "Dallas (us-south)" selected. A blue arrow labeled "1" points to this dropdown.

Select a pricing plan: A table lists available plans. The "Lite" plan is selected, showing a price of "Free".

Plan	Features	Pricing
Lite	Service instance 20 capacity unit-hours (CUH) included: Capacity Type: <ul style="list-style-type: none">• 1 vCPU and 4 GB RAM = 0.5 capacity units required per hour• 2 vCPU and 8 GB RAM = 1 capacity units required per hour• 4 vCPU and 16 GB RAM = 2 capacity units required per hour• 8 vCPU and 32 GB RAM = 4 capacity units required per hour• 16 vCPU and 64 GB RAM = 8 capacity units required per hour Auto AI <ul style="list-style-type: none">• 8 vCPU and 32 GB RAM = 20 capacity units required per hour Auto AI with Joined Data <ul style="list-style-type: none">• 1 Driver: 2 vCPU and 8 GB RAM + 1 Executor: 2 vCPU and 8 GB RAM = 10 capacity units required per hour, plus 5 capacity units per hour for each additional executor. After data is joined, CUH is consumed at a rate of 20 capacity units per hour. Decision Optimization <ul style="list-style-type: none">• 2 vCPU and 8 GB RAM = 30 capacity units required per hour• 4 vCPU and 16 GB RAM = 60 capacity units required per hour• 16 vCPU and 64 GB RAM = 60 capacity units required per hour Maximum 2 parallel Decision Optimization batch jobs per deployment Default number of deployment jobs retained per deployment space: 100 <p>The lite plan instance of the IBM Watson Machine Learning service provides you with a 20 capacity unit-hours per month during which models can be trained, evaluated, deployed, and scored.</p> <p>Lite plan services are deleted after 30 days of inactivity.</p> <td>Free</td>	Free

License Agreement: A checkbox labeled "I have read and agree to the following license agreements:" is checked. A blue arrow labeled "2" points to this checkbox.

Create Button: A blue button labeled "Create" is visible. A blue arrow labeled "3" points to this button.

Summary Panel: On the right, a "Summary" panel shows details for "Machine Learning" (Free), including location (Dallas), plan (Lite), service name (Machine Learning-g3), and resource group (Default).

At creating the “Machine Learning Service” select closet location, accept “license agreement” and click on “Create”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

24 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models*: Observe the “Model” created
In the “Models” section, at the “*Watson Machine Learning*” click on “*Heart_SPSS_C5-0*” to see the details of the model

24. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models: Observe the “Model” created

The screenshot shows the IBM Watson Studio interface. At the top, there is a navigation bar with 'IBM Cloud Pak for Data', 'All', a search bar, and an 'Upgrade' button. Below the navigation bar, there are tabs for 'Overview', 'Assets', 'Environments', 'Jobs', 'Access control', and 'Settings'. The 'Assets' tab is active, and a search bar is present with the text 'What assets are you looking for?'. The main content area is divided into two sections: 'Data assets' and 'Models'. The 'Data assets' section shows a table with columns 'Name', 'Type', 'Created by', and 'Last modified'. It lists five CSV files: 'Heart_disease.csv', 'cluster1.csv', 'cluster2.csv', and 'diabetes.csv'. The 'Models' section is titled 'Watson Machine Learning models' and contains a table with columns 'Name', 'Type', 'Software specification', and 'Last modified'. A blue arrow points to the 'Heart C5-0' model entry. Below the 'Models' section, there is a 'Modeler flows' section with a table showing a flow named 'Heart_SPSS'.

Name	Type	Created by	Last modified
Heart_disease.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 05:43 PM
cluster1.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 05:21 PM
cluster2.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 05:21 PM
diabetes.csv	Data Asset	Research Consultant Services	Oct 08, 2021, 12:40 PM

Name	Type	Software specification	Last modified
Heart C5-0	spss-modeler_18.2	spss-modeler_18.2	Oct 12, 2021

Name	Type	Created by	Last modified
Heart_SPSS	SPSS Modeler	Research Consultant Services	Oct 12, 2021, 10:48 AM

In the “Models” section, at the “*Watson Machine Learning Models*” click on “*Heart_SPSS_C5-0*” to see the details of the model

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

25 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS_C5-0 "Machine Learning Model "Overview"* Click "Overview" to see the model: summary, Input and Output Schema, then click in "Promote to deployment space"

25. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Heart_SPSS_C5-0 "Machine Learning Model "Overview"*

Model: Heart_SPSS_C5-0

Overview Activities

Summary

Model Type	spss-modeler_18.2
Software specification	spss-modeler_18.2
Training date	2 Dec 2020, 5:30 PM

Input Schema

Column	Type
AGE	"integer"
CA	"string"
CHOL	"integer"
CP	"integer"
EXANG	"integer"
FBS	"integer"
NUM	"integer"
OLDPEAK	"double"
RESTECG	"integer"

Output Schema

Column	Type
\$C-NUM	"integer"
\$CC-NUM	"double"
AGE	"integer"
CA	"string"
CHOL	"integer"
CP	"integer"
EXANG	"integer"
FBS	"integer"
NUM	"integer"

Click "Overview" to see the model: summary, Input and Output Schema, then click in "Promote to deployment space"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

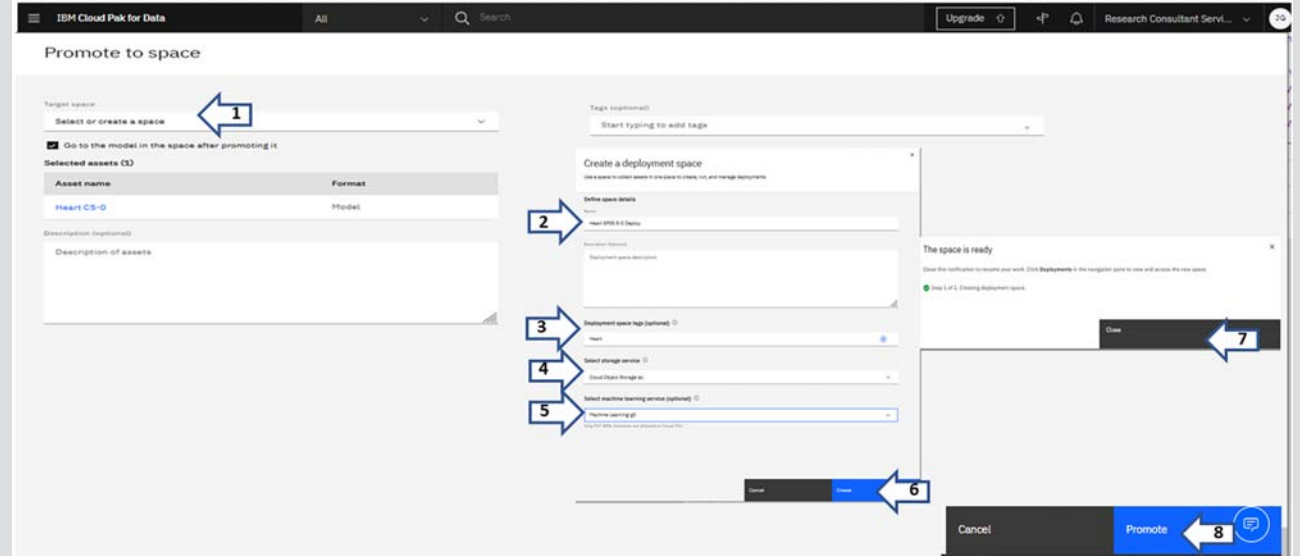
(Continued)

Slide Description

Screen figure

26 IBM Watson Studio screen: My Projects/AI and Cognitive Models/Heart_SPSS_C5-0 "Machine Learning Model add deployment"
In the ML model for "Heart_SPSS_C5-0," click target space and select "Create a new deployment space". A new request name = "Heart SPSS 5-0 deploy", tag = "Heart", storage service = "your assign storage service", Machine Learning service = "your assign ML service", click on "Create" and wait to the space to be created, and close. Finally, click on "Promote"

26. IBM Watson Studio screen: My Projects/AI and Cognitive Models/Heart_SPSS_C5-0 "Machine Learning Model add deployment"



In the ML model for "Heart_SPSS_C5-0", click Target space and select "Create a new deployment space". A new request name= "Heart SPSS 5-0 deploy", tag="Heart", storage service="your assign storage service", Machine Learning service="your assign ML service", click on "Create" and wait to the space to be created, and close. Finally, click on "Promote"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

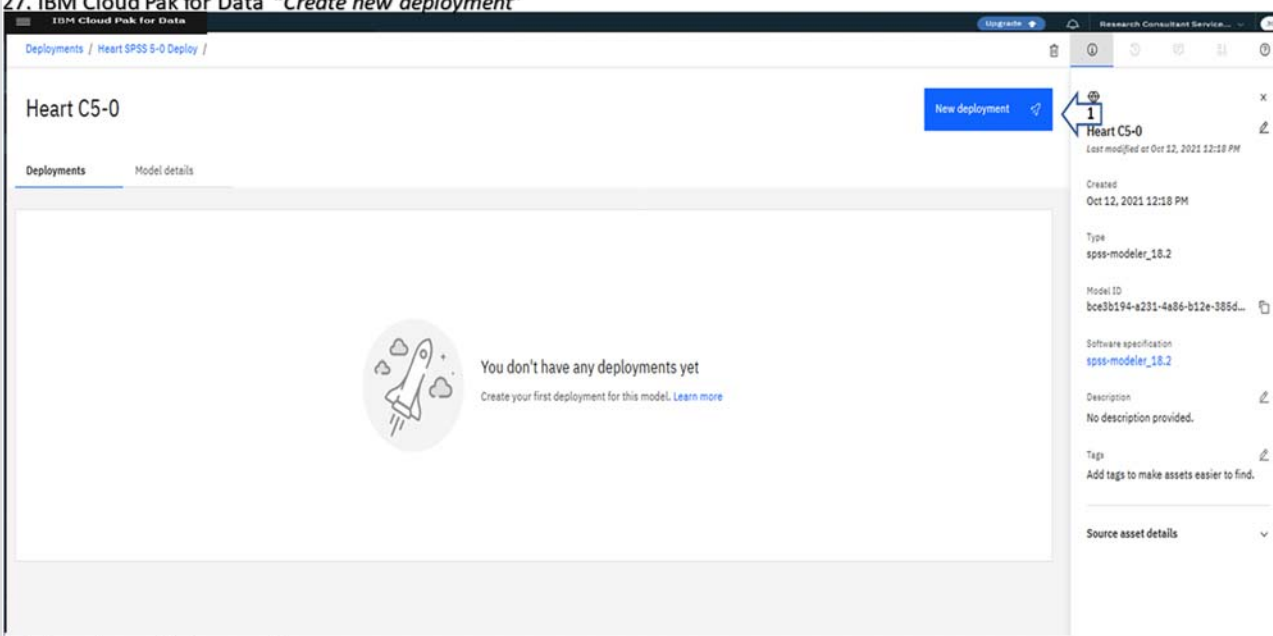
(Continued)

(Continued)

Slide Description
27 IBM Cloud Pak for Data
"Create new deployment"
Click on "New deployment"
button

Screen figure

27. IBM Cloud Pak for Data "Create new deployment"



The screenshot shows the IBM Cloud Pak for Data interface. The main content area displays a message: "You don't have any deployments yet. Create your first deployment for this model. [Learn more](#)". A blue "New deployment" button is visible in the top right corner. The sidebar on the right shows details for the "Heart C5-0" model, including creation date, type, model ID, and software specification.

Click on "New deployment" button

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

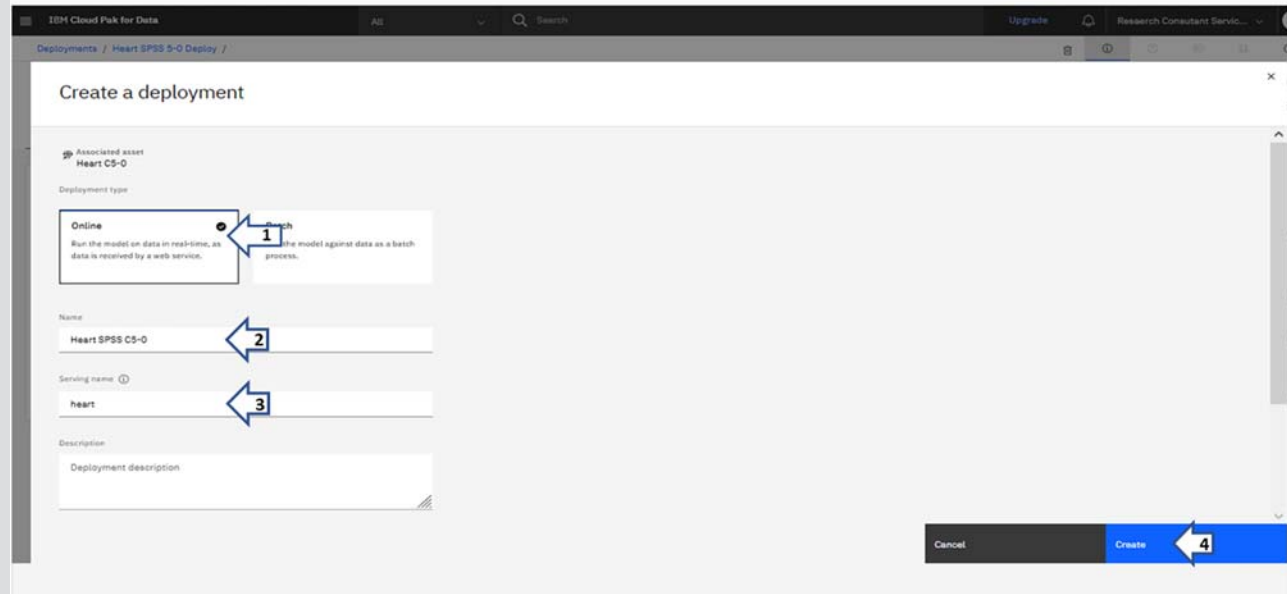
(Continued)

Slide Description

28 IBM Watson Studio screen: Deployment/spaces and Deployment/Heart_SPSS_5.0
At Create a deployment windows, select deployment type = "Online", name = "Heart SPSS C5-0", service name = "heart" and click on "Create"

Screen figure

28. IBM Watson Studio screen: Deployment/ spaces and Deployment/ Heart_SPSS_5.0



At Create a deployment windows, select deployment type="Online", name="Heart SPSS C5-0", service name="heart" and click on "Create"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 29
Description
IBM Watson Studio screen:
Create a deployment Online
Create a deployment type
"Online," Name = "Heart,"
description = "Heart_SPSS_C5-
0" and click "Create"

Screen figure

29. IBM Watson Studio screen: Create a deployment Online

Create a deployment

Associated asset
Heart_SPSS_C5-0

Deployment type

Online
Run the model on data in real-time, as data is received by a web service.

Batch
Run the model against data as a batch process.

Name
HEART

Description
HEART_SPSS_C5-0 on line

Cancel Create

Heart_SPSS_C5-0
Created Dec 2, 2020 11:10 AM
Type SPSS-modeler_18.2
Model ID 509d100c-702e-4c9e-8550-09d4...
Software specification SPSS-modeler_18.2
Description NO description provided.

Create a deployment type "Online", Name="Heart", Description="Heart_SPSS_C5-0" and click "Create"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

30 IBM Watson Studio screen:
Deployment Online
Select the deployment
"Online," click in "Heart SPSS
C5-0"

30. IBM Watson Studio screen: Deployment Online

The screenshot displays the IBM Watson Studio interface for the 'Heart SPSS C5-0' deployment. At the top, there is a navigation bar with 'IBM Cloud Pak for Data', 'All', a search bar, and an 'Upgrade' button. Below this, the breadcrumb path is 'Deployments / Heart SPSS C5-0 Deploy /'. The main content area is titled 'Heart C5-0' and has tabs for 'Deployments' and 'Model details'. A notification banner at the top right indicates 'Online deployment ready' for the 'Heart SPSS C5-0' deployment. The 'Deployments' tab shows a table with columns for 'DEPLOYMENT TYPES', 'Name', 'Status', and 'Last modified'. The 'Online' deployment type is selected, and the 'Heart SPSS C5-0' deployment is listed with a status of 'Deployed'. Two blue arrows with numbers 1 and 2 point to the 'Online' deployment type and the 'Heart SPSS C5-0' deployment respectively. A sidebar on the right shows details for the selected deployment, including 'Type: spss-modeler_18.2', 'Model ID: bce3b194-a231-4a86-b12e-385d...', 'Software specification: spss-modeler_18.2', and 'Description: No description provided.' Below the table, there is a text instruction: 'Select the deployment "Online", click in "Heart SPSS C5-0"'. At the bottom of the screenshot, there is a footer: 'From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa'.

DEPLOYMENT TYPES	Name	Status	Last modified
Online	Heart SPSS C5-0	Deployed	Oct 12, 2021 12:37 PM
Batch	(0)		

Select the deployment "Online", click in "Heart SPSS C5-0"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

31 IBM Watson Studio screen:
Copy the “API reference > Direct Link > Endpoint”
Copy the “API reference > Direct Link > Endpoint” in notepad and save it as “... \Exercises_book_ABME \CH4\SPSS_MODELER \SPSS_Classifiers \IBM_Cloud_Shell.txt” for later use. Observe the “Code Snippets” for: “cURL,” “Java,” “JavaScript,” “Python,” and “Scala.” Finally select the “Test” tab

31. IBM Watson Studio screen: Copy the “API reference > Direct Link > Endpoint”

The screenshot displays the IBM Watson Studio interface for a deployment named "Heart SPSS C5-0". The interface is divided into several sections:

- API reference:** A tab labeled "Test" is selected, indicated by a blue arrow with the number "3".
- Direct link:** A section titled "Direct link" containing an "Endpoint" field with two URLs:
 - https://us-south.ml.cloud.ibm.com/ml/v4/deployments/heart/predictions?version=2021-10-12
 - https://us-south.ml.cloud.ibm.com/ml/v4/deployments/920b6def-452d-47fe-9868-3a56c3342956/predictions?version=2021-10-12Blue arrows with numbers "1" and "2" point to these URLs.
- Code snippets:** A section titled "Code snippets" with tabs for "cURL", "Java", "JavaScript", "Python", and "Scala". The "cURL" tab is active, showing a curl command and instructions for using an API key and SAML token.
- Right sidebar:** A panel showing deployment details for "Heart SPSS C5-0", including creation and update times, deployment ID, software specification, and serving name.

Copy the “API reference > Direct Link > Endpoint” in notepad and save it as “... \Exercises_book_ABME \CH4\SPSS_MODELER \SPSS_Classifiers \IBM_Cloud_Shell.txt” for later use. Observe the “Code Snippets” for: “cURL,” “Java,” “JavaScript,” “Python” and “Scala”. Finally select the “Test” tab.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

32 IBM Watson Studio screen: Copy the "Testing the model" In test form enter the Input data: "Age = 67, SEX = 1, CO = 1, TESTPBS = 160, CHOL = 286, FBS = 0, RESTECG = 2, THALACH = 108, EXANG = 1, OLDPEAK = 1.5, SLOPE = 2, CA = 3, THAL = 3, NUM = 1," press icon "Add to list," then click on "Predict" button, the result "for num is Indicated as "\$SCC-NUM = 0.8108." Finally, click on the top right menu and make "Log out"

32. IBM Watson Studio screen: Copy the "Testing the model"

The screenshot shows the IBM Watson Studio interface for a Heart SPSS C5-0 model. The 'Enter input data' section contains a table with the following values: AGE=67, SEX=1, CP=1, TESTPBS=160, CHOL=286, FBS=0, RESTECG=2, THALACH=108, EXANG=1, OLDPEAK=1.5, SLOPE=2, CA=3, THAL=3, NUM=1. A blue callout '1' points to the AGE field, '2' to the 'Add to list' button, '3' to the OLDPEAK field, and '4' to the NUM field. A blue callout '5' points to the 'Predict' button. The 'Predictions' section shows a JSON output with a 'values' array containing the predicted SCC-NUM value of 0.8108108108108108. A blue callout '6' points to the 'values' array, and '7' points to the SCC-NUM value. In the top right corner, a user profile menu for 'Jorge Garza-Ulloa' is visible, with a blue callout '8' pointing to the user name and '9' pointing to the 'Log out' option. A blue callout 'Optional' points to a menu icon in the top left of the input data section.

In test form enter the Input data: "Age=67, SEX=1, CP=4, TESTPBS=160, CHOL=286, FBS=0, RESTECG=2, THALACH=108, EXANG=1, OLDPEAK=1.5, SLOPE=2, CA=3, THAL=3, NUM=1", press icon "Add to list", then click on "Predict" button, the result "for num is Indicated as "\$SCC-NUM=0.8108". Finally, click on the top right menu and make "Log out"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

33 IBM Watson Studio screen: Testing deployment of Heart in <https://cloud.ibm.com/shell/>, Step 1) Variable SCORING_ENDPOINT STEP 1) variable SCORING_ENDPOINT: Follow the instruction to complete the first command line to be used at CLOUD IBM Shell to define the variable "SCOREPOINT" as indicated in the top screen. When ready, copy and paste it as shown in the lower screen

33. IBM Watson Studio screen: Testing deployment of Heart using <https://cloud.ibm.com/shell/>: Step 1) Build SCORING ENDPOINT



STEP 1) Variable SCORING_ENDPOINT: Follow the instruction to complete the first command line to be used at CLOUD IBM Shell to define the variable "SCOREPOINT" as indicated in the top screen. When is ready, copy and paste it as shown in the lower screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

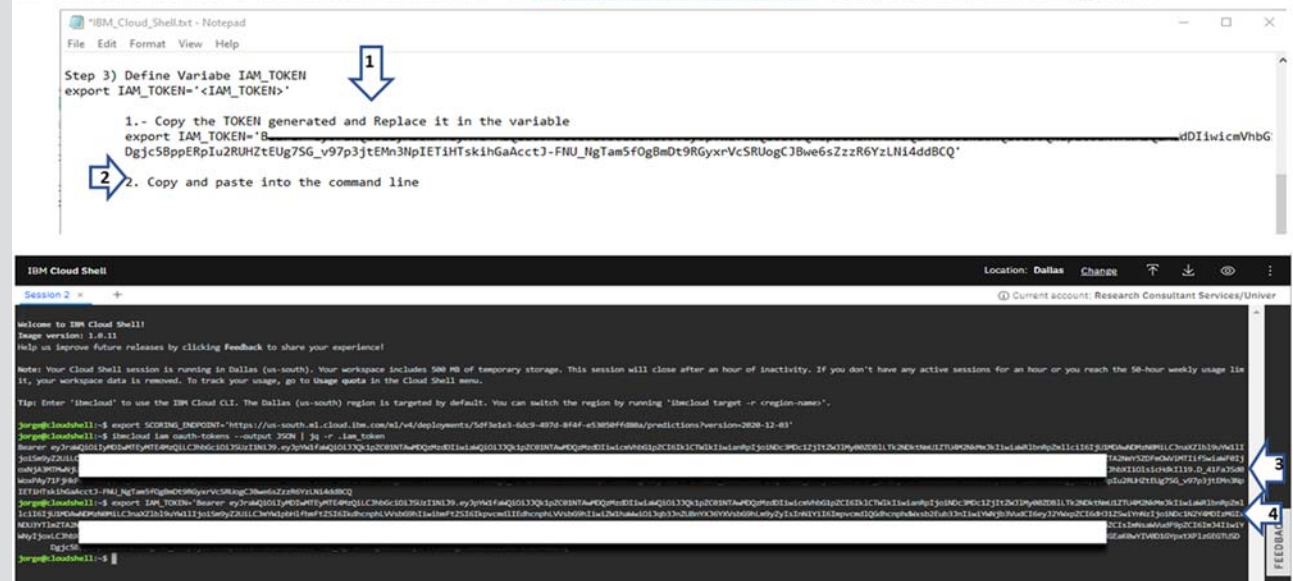
(Continued)

Slide **Description**

35 Go to the browser to use the IBM Cloud Shell at <https://cloud.ibm.com/shell>, Step 3) Define variable IAM_TOKEN: Copy the TOKEN generated in the last slide, and paste in the command as shown in the upper screen, then copy the entire command into the IBM Cloud Shell as shown in the lower screen

Screen figure

35. Go to the browser to use the IBM Cloud Shell at <https://cloud.ibm.com/shell> Step 3) Define variable IAM_TOKEN



Step 3) Define variable IAM_TOKEN : Copy the TOKEN generated in the last slide, and paste in the command as shown in the upper screen, then copy the entire command into the IBM Cloud Shell as shown in the lower screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

Conclusions

We can have the following conclusions for obtaining “ML model using Supervised Learning with a Decision Tree algorithms for a heart diseases dataset.” “C&R Tree” and “C5.0” generated different models for the “Heart_disease.csv” dataset, both were analyzed based on their “Coincide Matrices” and “Performance Evaluation.” The “C5.0” obtained 79.21% of correct classification versus “C&R Tree” with only 63.7%. Thus “C5.0” is better and it can be deployed and tested and ready to be used as a web service. The use of the “IBM Cloud shell” similar to “cURL” used on chapter 2, section 8.2.3.3 Developing IBM Cloud NLP applications with IBM APIs

Recommendations

- A larger dataset will help to analyze with more precision the “NUM” with predicted value, for a diagnosis of heart disease.
- Apply more “ML models for classification and/or regression” and decide which is has better performance best for this kind of dataset.

4.12.2.3 Research tutorial 4.3 IBM Watson SPSS Modeler Flow for “Kidney disease ML Auto Classifiers Models and deploy the best model”

4.12.2.3.1 Case for research

“Obtain ML models using Auto Classifiers algorithms available at IBM Watson SPSS Modeler Flow for analyzing a Kidney diseases dataset, and deploy the best model.”

4.12.2.3.2 General objective for this research

“Use IBM Watson Studio IBM SPSS Modeler Flow” to obtain an ML models from Auto Classifiers algorithms for a kidney disease dataset and deploy the best model.

4.12.2.3.3 Background for “Kidney disease”

“Chronic kidney disease,” or “chronic kidney failure,” describes the gradual loss of “kidney function.” The “kidneys” filter wastes and excess fluids from blood, which are then excreted in your urine. When “chronic kidney disease” reaches an advanced stage, dangerous levels of fluid, electrolytes and wastes can build up in the human body [23].

Factors that may increase the risk of “chronic kidney disease” include:

- “Diabetes,”
- “High blood pressure,”
- “Heart and blood vessel (cardiovascular) disease,”
- “Smoking,”
- “Obesity,”

- “Race, such as being African American, Native American, or Asian American,”
- “Family history of kidney disease,”
- “Abnormal kidney structure,” and
- “Older age.”

4.12.2.3.4 Specific objectives

Use “Auto Classifier to obtain models for the Kidney_disease.csv” dataset. The specific objectives are:

- Predict when patient has “Chronic kidney disease” based on the attributes of a given dataset.
- Use IBM Watson SPSS Modeler “Auto Classifier”
- Evaluate the “ML model” for the “ML Auto Classifier algorithms,”
- Choose the best “ML Auto Classifier Model” based on:
 - “Model evaluation accuracy,”
 - “Coincide matrices,”
 - “Performance evaluation.”
- Save the “ML model stream” to be used when the deployment job is run
- Save “branch as a model,”
- Create deployment and test it on a “JSON” form.

4.12.2.3.5 Dataset

The dataset “Kidney_disease.csv*” is based on information from two hospitals over a 2-month period of chronic kidney disease, available at the UCI Machine Learning Repository [24]. This database contains 25 attributes, from 400 instances (patients). The goal field is “class,” which refers to the presence of chronic kidney disease in the patient as “ckd” or not presence as “notckd.” The “14 fields (attributes)” of this dataset, their descriptions and values are indicated in Table 4.2.

4.12.2.3.6 Procedure

The steps to obtain “ML Auto Classifiers Models for a chronic kidney diseases dataset, and deploy the best model” are summarized in Table of slides 4.3 where each step of the example is visually explained using screen sequences with instructions.

Note: The “IBM Cloud website” is continuously evolving every day by consequence the IBM CLOUD website screens change frequently, some screens could be updated to optimize and simplify the procedures. I recommend understanding very well the objectives, applying them accordingly with the new screen’s formats shown at this table and the current IBM Cloud website contents. This research uses the “IBM Cloud Pak for Data” which is a fully integrated data and AI Watson platform.

TABLE 4.2 Dataset “*Kidney_disease.csv*” fields and descriptions.

Field	Description * 400 instances of patients Missing attribute values are denoted by “?”
age	Age in years (numerical)*
bp	Blood pressure in mm/Hg (numerical)
sg	Specific gravity (nominal) = [1.005,1.010,1.015,1.020,1.025]
al	Albumin (nominal) = [0,1,2,3,4,5]
su	Sugar (nominal) = [0,1,2,3,4,5]
rbc	Red blood cells (nominal) = ['normal', 'abnormal']
pc	Pus cell (nominal) = ['normal', 'abnormal']
pcc	Pus cell clumps(nominal) = ['present', 'not present']
ba	Bacteria(nominal) = ['present', 'notpresent']
bgr	Blood glucose random in mg/dL (numerical)
bu	Blood urea in mg/dL (numerical)
sc	Serum creatinine in mg/dL (numerical)
sod	Sodium in mEq/L (numerical)
pot	Potassium in mEq/L (numerical)
hemo	Hemoglobin in gm (numerical)
pcv	Packed cell volume (numerical)
wc	White blood cell count in cells/cmm = cells per cubic millimeter (numerical)
rc	Red blood cell count in millions/cmm = cells per cubic millimeter (numerical)
htn	Hypertension(nominal) = ['yes', 'no']
dm	Diabetes mellitus (nominal) = ['yes', 'no'] *type 1 or type 2 diabetes
cad	Coronary artery disease (nominal) = ['yes', 'no']*
appet	Appetite(nominal) = ['good', 'poor']
pe	Pedal edema(nominal) = ['yes', 'no']
ane	Anemia(nominal) = ['yes', 'no']
class	Class (nominal) = ['ckd', 'notckd']

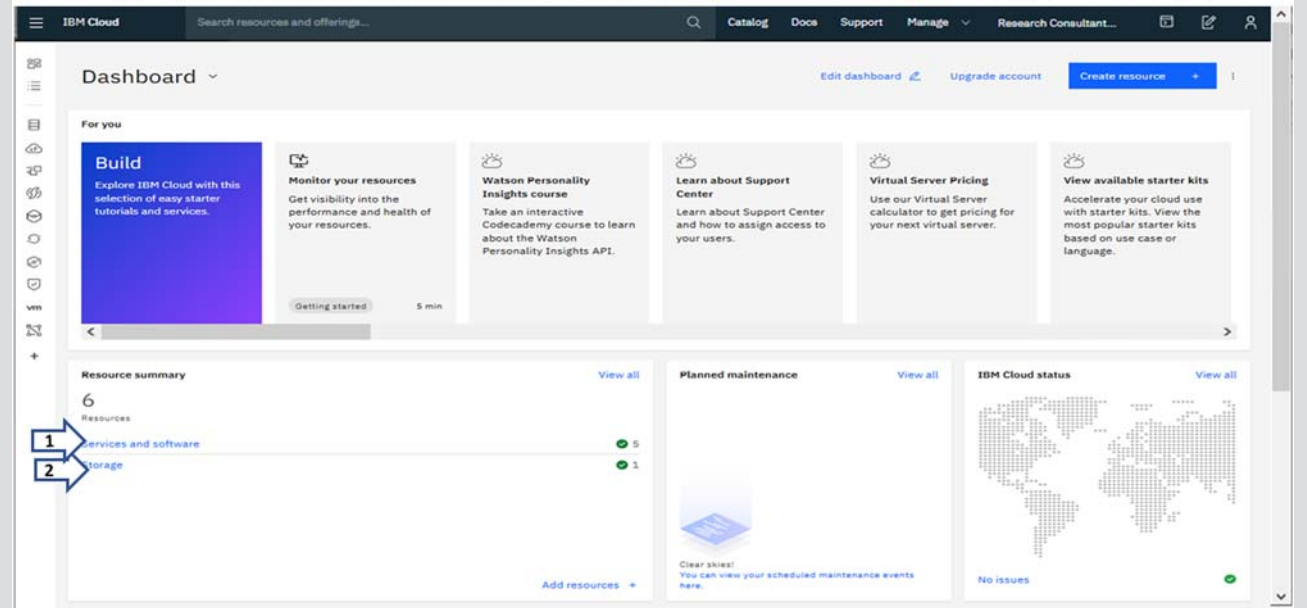
Note*: This dataset is available in the companion directory of the book, in the following directory: “...Exercises_book_ABME\CH4\SPSS_MODELER\SPSS_AutoClassifier\Kidney_disease.csv”.

Table of slides 4.3 steps to obtain ML models using Auto Classifiers algorithms available at IBM Watson SPSS Modeler Flow for a Kidney diseases dataset, and deploy the best model.

Slide Description Screen figure

1 Login to your IBM Cloud Account at the website: <https://cloud.ibm.com/login> entering your assigned IBMid and Password
 Note: You must have the 5 services available and 1 storage created as per Chapter 3 and Chapter 4, then click the "Services" to list them

1. Make login in your account IBM Cloud Account in the website: <https://cloud.ibm.com/login> entering your IBMid and Password



You must have the 5 services available, and 1 storage created on Chapter 3 and Chapter 4, then click on "Services" to list them

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 2 Description In the IBM Cloud—Resource list verifying the services Click to expand “Services(5)” and “Storage(1),” check the availability of the “Machine Learning service” necessary for this research and click the service “Watson Studio” to go to its service screen

Screen figure

2. In the IBM Cloud – Resource list verifying the services

Name	Group	Location	Product	Status	Tags
Machine Learning-g3	Default	Dallas	Machine Learning	Active	
Natural Language Understanding-vw	Default	Dallas	Natural Language Understan...	Active	nlp
Speech to Text-aj	Default	Dallas	Speech to Text	Active	nlp
Text to Speech-uj	Default	Dallas	Text to Speech	Active	
Watson Studio-2p	Default	Dallas	Watson Studio	Active	

Click to expand “Services(5)” and “Storage(1)”, check the availability of the Machine Learning service necessary for this tutorial and click the service “Watson Studio” to go to its service screen

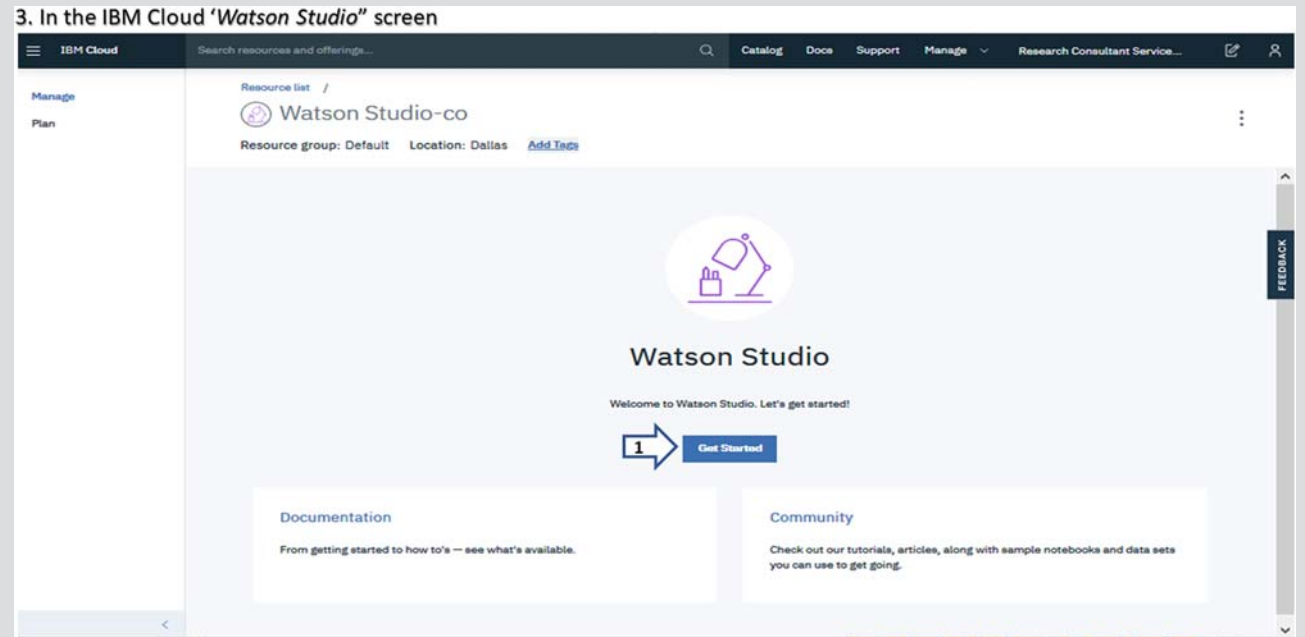
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description
3 In the IBM Cloud "Watson Studio"
screen
Note: Press the button "Get Started"

Screen figure



Press the button "Get Started"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

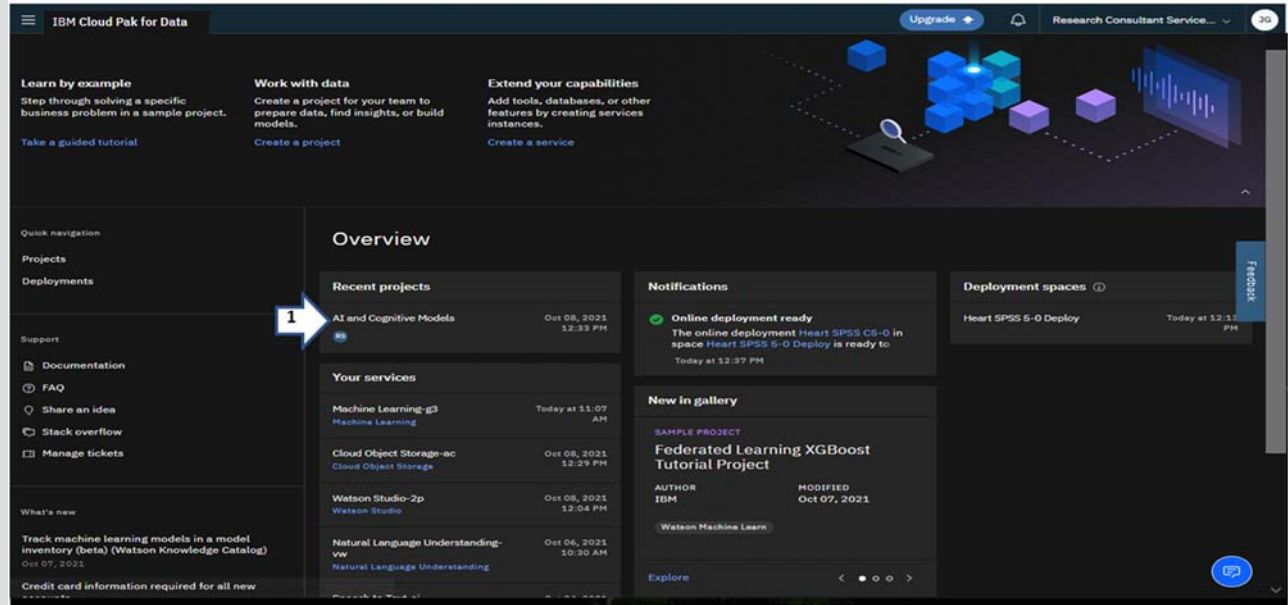
(Continued)

(Continued)

Slide 4
Description
In the IBM Watson Studio welcome screen
Click the recently updated project
"AI and Cognitive Models"

Screen figure

4. In the IBM Watson Studio welcome screen



Click the recently updated project "AI and Cognitive Models"

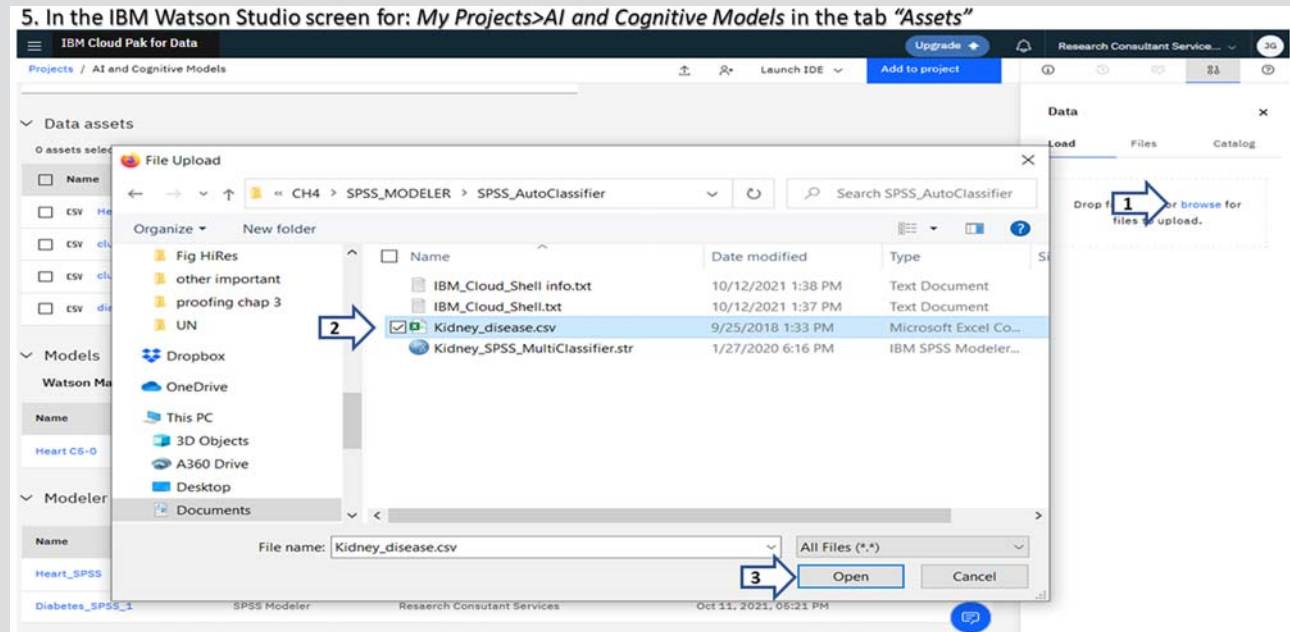
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description Screen figure

5 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models* in the tab "Assets" Select New Dataset and load asset using "browse," select the dataset "Kidney_disease.csv" in the data companion directory and press the button "Open"



Select New Dataset and load asset using "browse", select the data set "Kidney_disease.csv" in the data companion directory and press the button "Open"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

6 In the IBM Watson Studio *My Projects / AI and Cognitive Models* section Assets: observe that *Kidney_disease.csv* is available
Click on the button *Add to project*

6. IBM Watson Studio “My Projects / AI and Cognitive Models” section Assets: observe that asset “Kidney_disease.csv” is available

The screenshot displays the IBM Watson Studio interface for the 'My Projects / AI and Cognitive Models' section. The 'Data assets' section is expanded, showing a table of assets. The first asset, 'Kidney_disease.csv', is highlighted with a red arrow labeled '1'. A red arrow labeled '2' points to the 'Add to project' button in the top right corner. Below the data assets, the 'Modeler flows' section is also expanded, showing a table of modeler flows. The 'Heart_SPSS' flow is highlighted. A 'Data' panel is visible on the right side of the interface, showing a file upload area with the text 'Drop files here or browse for files to upload.' and a 'Load' button.

Name	Type	Created by	Last modified
<input type="checkbox"/> CSV Kidney_disease.csv	Data Asset	Research Consultant Services	Oct 12, 2021, 06:14 PM
<input type="checkbox"/> CSV Heart_disease.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 06:43 PM
<input type="checkbox"/> CSV cluster1.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 06:21 PM
<input type="checkbox"/> CSV cluster2.csv	Data Asset	Research Consultant Services	Oct 11, 2021, 06:21 PM
<input type="checkbox"/> CSV diabetes.csv	Data Asset	Research Consultant Services	Oct 08, 2021, 12:40 PM

Name	Type	Software specification	Last modified
Heart_CS-0	spss-modeler_18.2	spss-modeler_18.2	Oct 12, 2021

Name	Type	Created by	Last modified
Heart_SPSS	SPSS Modeler	Research Consultant Services	Oct 12, 2021, 10:48 AM
Diabetes_SPSS_1	SPSS Modeler	Research Consultant Services	Oct 11, 2021, 06:21 PM

Observe in “Data assets” section the “Kidney_disease.csv” is loaded and click on the button “Add to project”

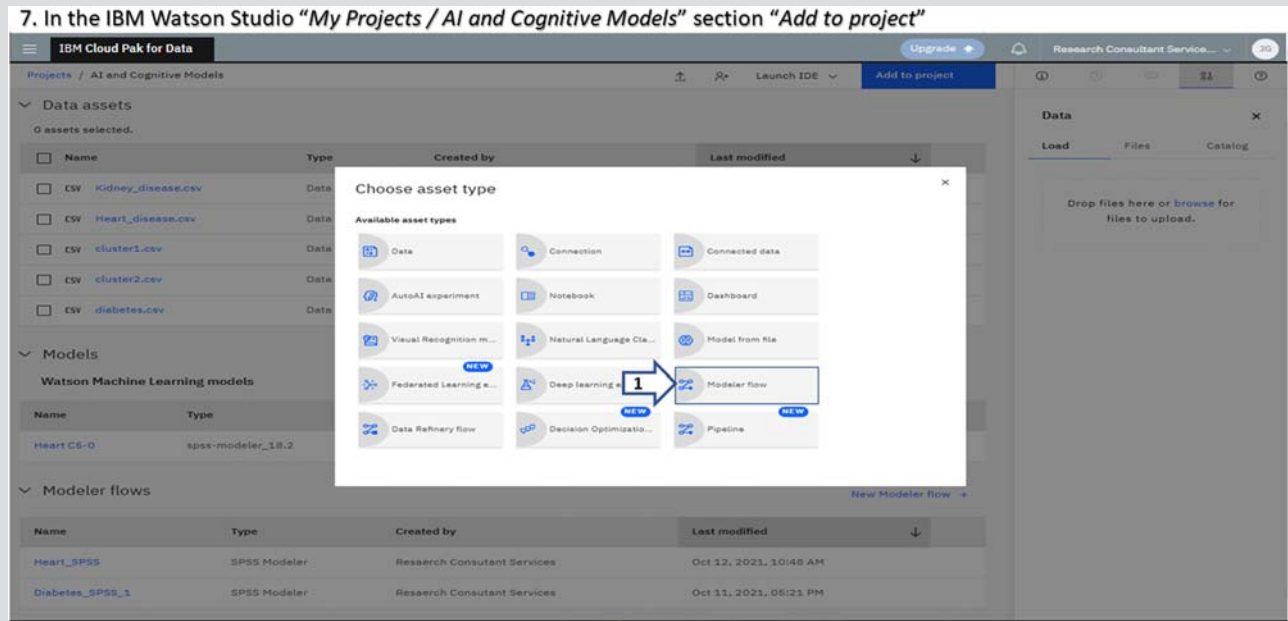
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description Screen figure

7 In the IBM Watson Studio "My Projects/AI and Cognitive Models" section "Add to project"
Click on the button "Modeler flow"



Click on the button "Modeler Flow"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

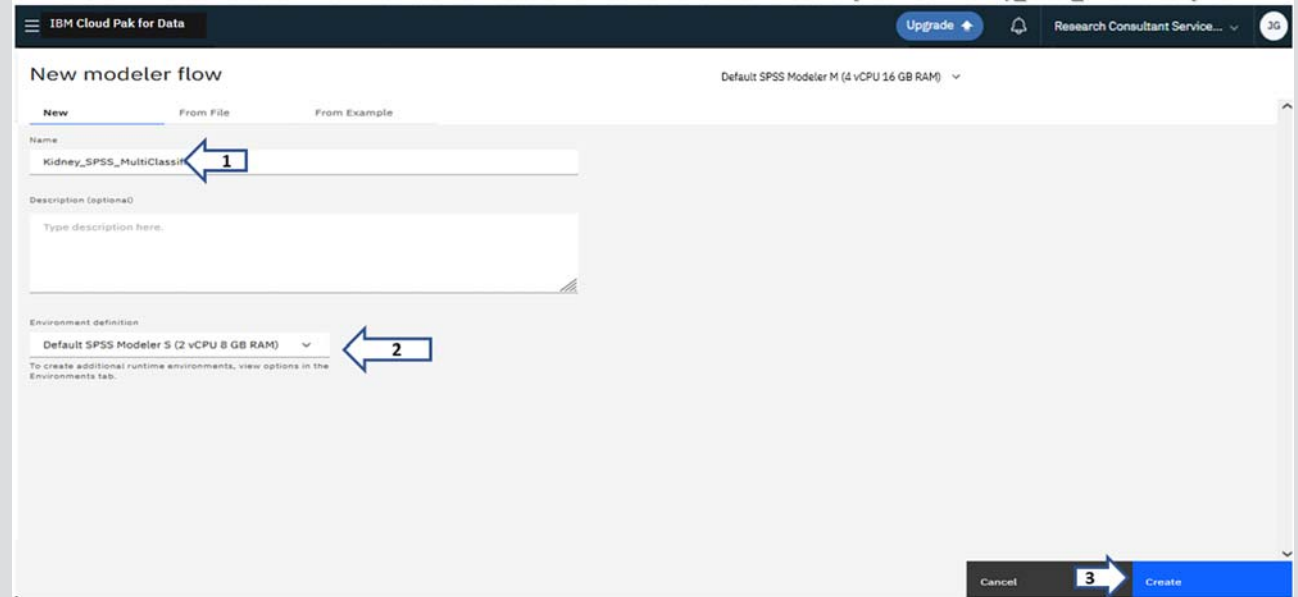
(Continued)

(Continued)

Slide 8 Description In the IBM Watson Studio—New modeler flow screen: enter the name of the “New modeler flow” as “Kidney_SPSS_MultiClassifiers” Select the options: Environment definition as “Default SPSS Modeler S (2 vCPU * GB RAM)” and click on the button “Create”

Screen figure

8. In the IBM Watson Studio – New modeler flow screen: enter name of the “modeler flow” as “Kidney_SPSS_MultiClassifiers”



Select the options: Environment definition as “Default SPSS Modeler S (2 vCPU * GB RAM)” and click the button “Create”

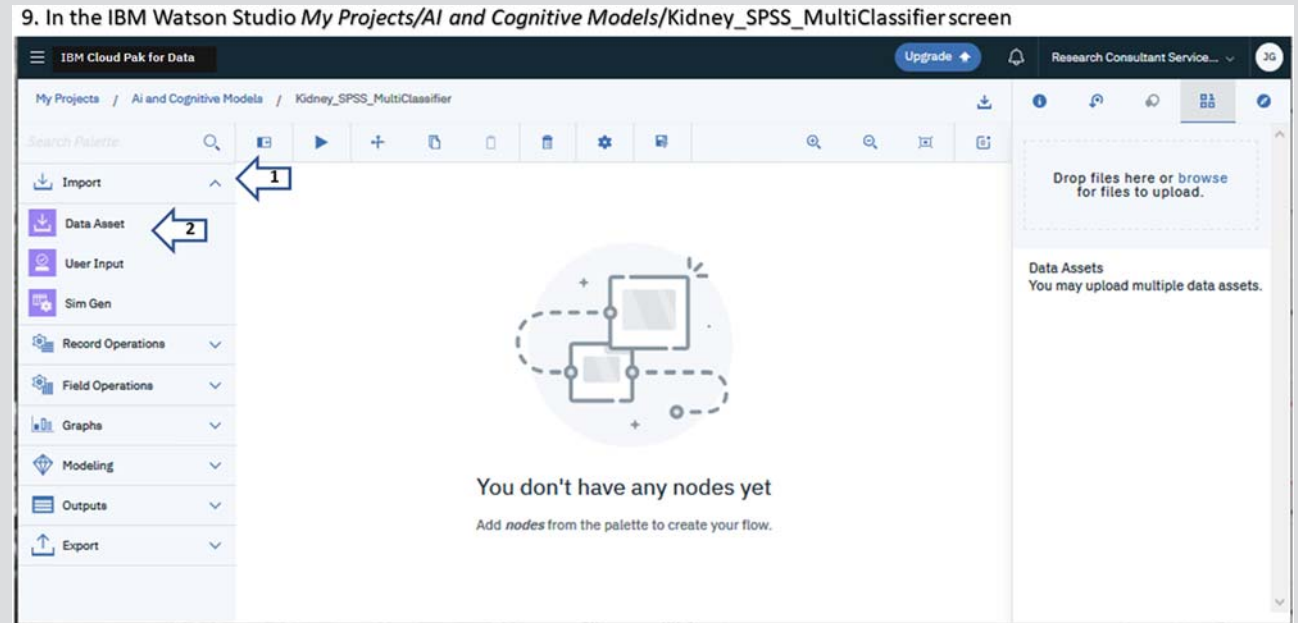
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 9 Description In the IBM Watson Studio "My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier screen" Select "Import" to expand its menu, click and drag "Data Asset" to create the first node of the model flow

Screen figure



Select "Import" to expand its menu, click and drag "Data Asset" to create the first node of the model flow

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

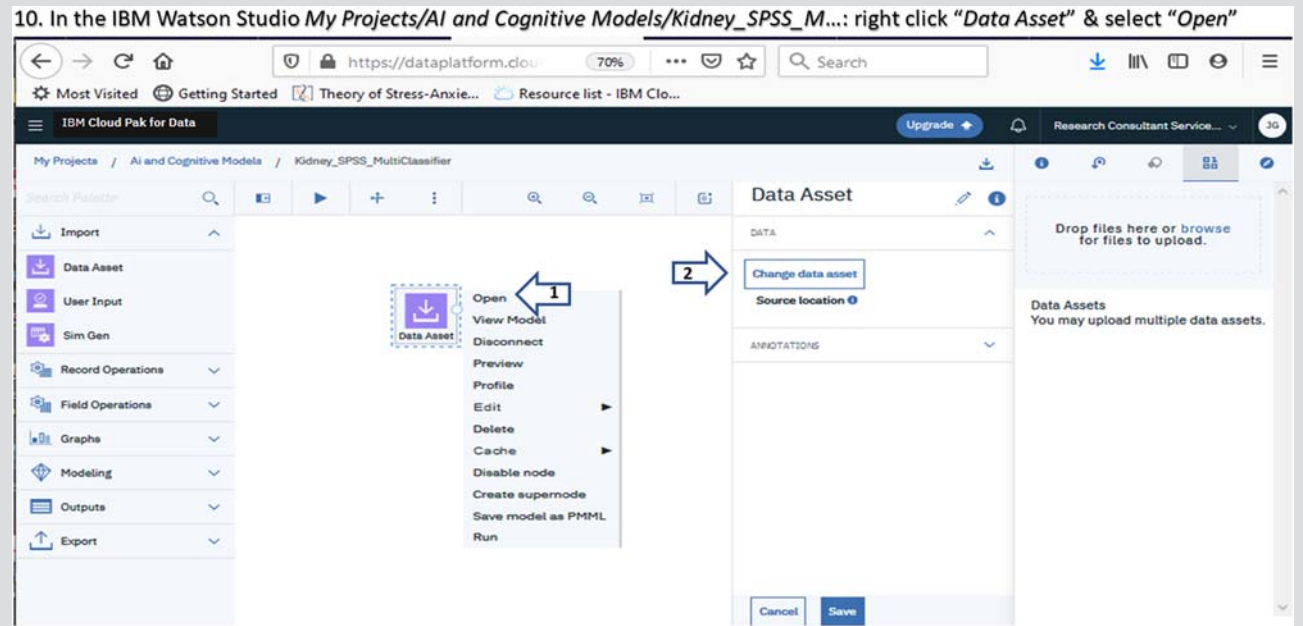
(Continued)

(Continued)

Slide **Description**

10 In the IBM Watson Studio "My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier" right click "Data Asset" and select "Open"
Click the button "Change data asset" as indicated in the slide

Screen figure



Click the button 'Change data asset' as indicated in the slide

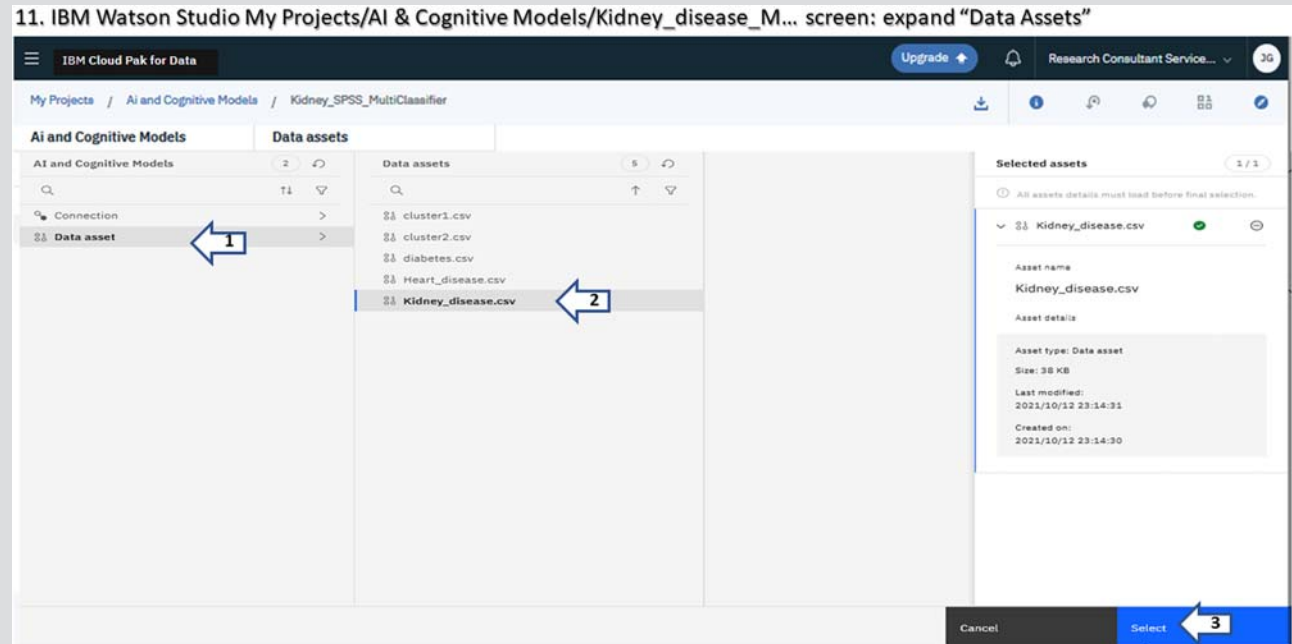
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 11 Description In the IBM Watson Studio “My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier” screen: observe “Data Assets” Select: “Data assets,” “Kidney_disease.csv” then press the button “Create”

Screen figure



Select: “Data assets”, “Kidney_disease.csv” then press the button “Create”

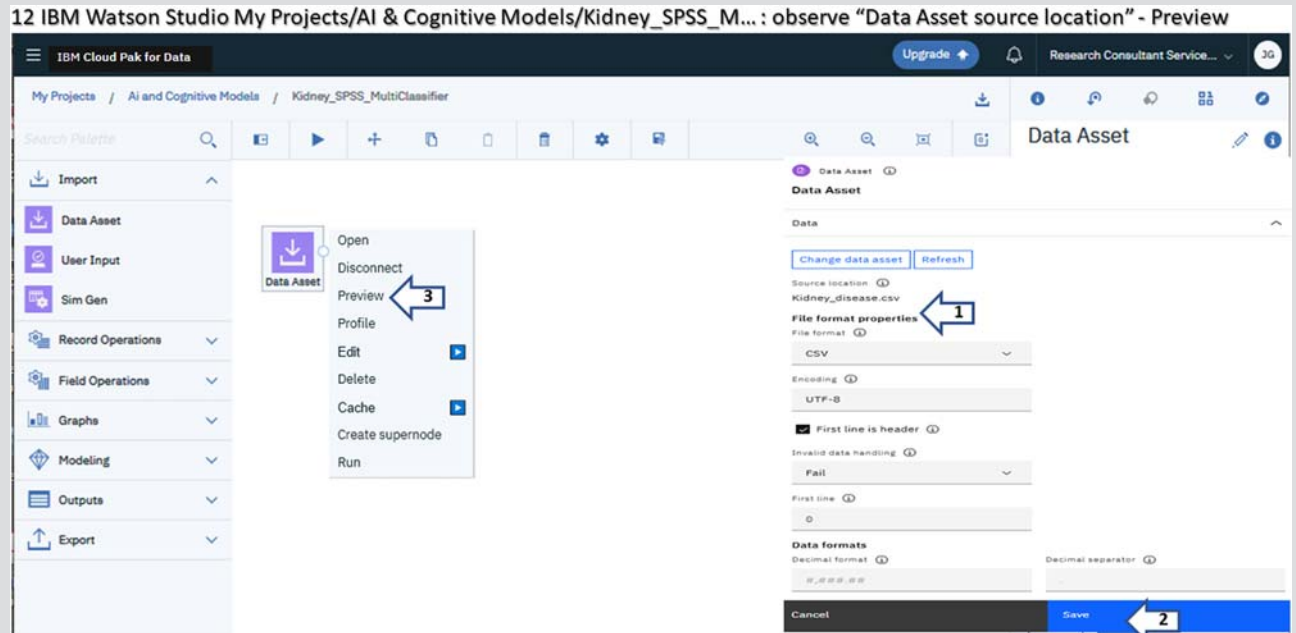
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 12 Description IBM Watson Studio "My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier" screen: observe Data Asset source location—Preview In "Data Asset" verify "Kidney_disease.csv", click on "Save". Make a right click on "data asset" and select "Preview"

Screen figure



In "Data Asset" verify "Kidney_disease.csv", click on "Save". Make a right click on data asset" and select "Preview"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

13 In the IBM Watson Studio screen: "My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier /Preview"
Observe in data tab, the column 'Class' is the assigned values from nominal range ['ckd', 'notckd'] if the patient has been previously diagnosed with "Kidneys disease" or not.
Note: Some field have "?" to indicate nonavailable values, return to SPSS Modeler select "Kidney_SPSS_MultiClassifiers" as shown in the slide

13. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier/Preview

	age	bp	sg	al	rbic	pc	pot	ba	lgr	ba	sc	sod	pot	hemo	pcv	wbc	rbc	hem	dm	cad	appet	ps	ane	class	
	Decimal	Decimal	String	String	String	String	String	String	String	String	Decimal	String	String	String	String	String	String	String	String	String	String	String	String	String	
1	48	80	1.02	1	?	normal	notpres...	notpresent	121	36	1.2	?	?	15.8	44	7500	5.2	yes	yes	no	good	no	no	ckd	
2	?	50	1.02	4	?	normal	notpres...	notpresent	?	18	0.6	?	?	11.9	36	6000	?	no	no	no	good	no	no	ckd	
3	42	80	1.05	2	3	normal	normal	notpres...	notpresent	423	53	1.6	?	?	6.6	31	7500	?	no	yes	no	poor	no	yes	ckd
4	48	70	1.005	4	0	normal	abnormal	present	notpresent	117	56	3.8	111	2.5	11.2	32	4700	3.9	yes	no	no	poor	yes	yes	ckd
5	51	80	1.01	2	0	normal	normal	notpres...	notpresent	106	26	1.4	?	?	11.6	35	7900	4.6	no	no	no	good	no	no	ckd
6	40	90	1.015	3	0	?	?	notpres...	notpresent	74	25	1.1	142	3.2	12.2	29	7600	4.4	yes	yes	no	good	yes	no	ckd
7	48	70	1.01	0	0	?	normal	notpres...	notpresent	100	54	24	104	4	12.4	36	?	?	no	no	no	good	no	no	ckd
8	52	100	1.015	3	0	normal	abnormal	present	notpresent	138	40	1.9	?	?	10.8	33	9600	4	yes	yes	no	good	no	yes	ckd
9	53	90	1.02	2	0	abnormal	abnormal	present	notpresent	70	107	7.2	114	3.7	9.5	29	12100	3.7	yes	yes	no	poor	no	yes	ckd
10	50	60	1.01	2	4	?	abnormal	present	notpresent	490	55	4	?	?	9.4	28	?	?	yes	yes	no	good	no	yes	ckd
11	43	70	1.01	3	0	abnormal	abnormal	present	notpresent	380	60	2.7	131	4.2	10.8	33	4500	3.8	yes	yes	no	poor	yes	no	ckd
12	48	70	1.015	3	1	?	normal	present	notpresent	208	72	2.1	138	5.8	9.7	28	12200	3.4	yes	yes	yes	poor	yes	no	ckd
13	48	70	?	?	?	?	?	notpres...	notpresent	86	86	4.6	135	3.4	9.8	?	?	?	yes	yes	yes	poor	yes	no	ckd
14	48	80	1.01	3	2	normal	abnormal	present	present	137	90	4.1	130	6.4	5.6	16	11000	2.6	yes	yes	yes	poor	yes	no	ckd
15	40	80	1.015	3	0	?	normal	notpres...	notpresent	76	142	9.6	141	4.9	7.6	14	3800	2.8	yes	no	no	good	no	yes	ckd
16	47	70	1.015	2	0	?	normal	notpres...	notpresent	99	44	2.3	138	4.1	12.4	?	?	?	no	no	no	good	no	no	ckd
17	47	80	?	?	?	?	?	notpres...	notpresent	114	87	5.2	139	3.7	12.1	?	?	?	yes	no	no	poor	no	no	ckd
18	60	100	1.025	0	3	?	normal	notpres...	notpresent	263	27	1.3	135	4.3	12.7	27	11400	4.3	yes	yes	yes	good	no	no	ckd
19	42	60	1.015	1	0	?	abnormal	present	notpresent	100	31	1.6	?	?	10.3	30	5300	3.7	yes	no	yes	good	no	no	ckd
20	41	80	1.015	2	0	abnormal	abnormal	notpres...	notpresent	173	148	3.9	135	5.2	7.7	14	9200	3.2	yes	yes	yes	poor	yes	yes	ckd
21	40	90	?	?	?	?	?	notpres...	notpresent	?	130	76	4.5	?	10.9	32	4300	1.6	yes	yes	yes	good	no	no	ckd
22	48	80	1.015	4	0	normal	abnormal	notpres...	notpresent	95	143	7.7	136	3.8	9.8	32	4900	3.4	yes	no	no	good	no	yes	ckd
23	42	100	1.015	4	0	normal	abnormal	notpres...	present	?	50	1.4	129	4	11.1	39	8200	4.6	yes	no	no	poor	no	no	ckd

Observe in data tab, the column 'Class' is the assigned values from nominal range ['ckd','notckd'] if the patient has been previously diagnosed with "Kidness disease". Note: Some field have "?" To indicate non-available values, return to "SPSS Modeler" select "Kidney_SPSS_MultiClassifiers" as shown in the slide

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

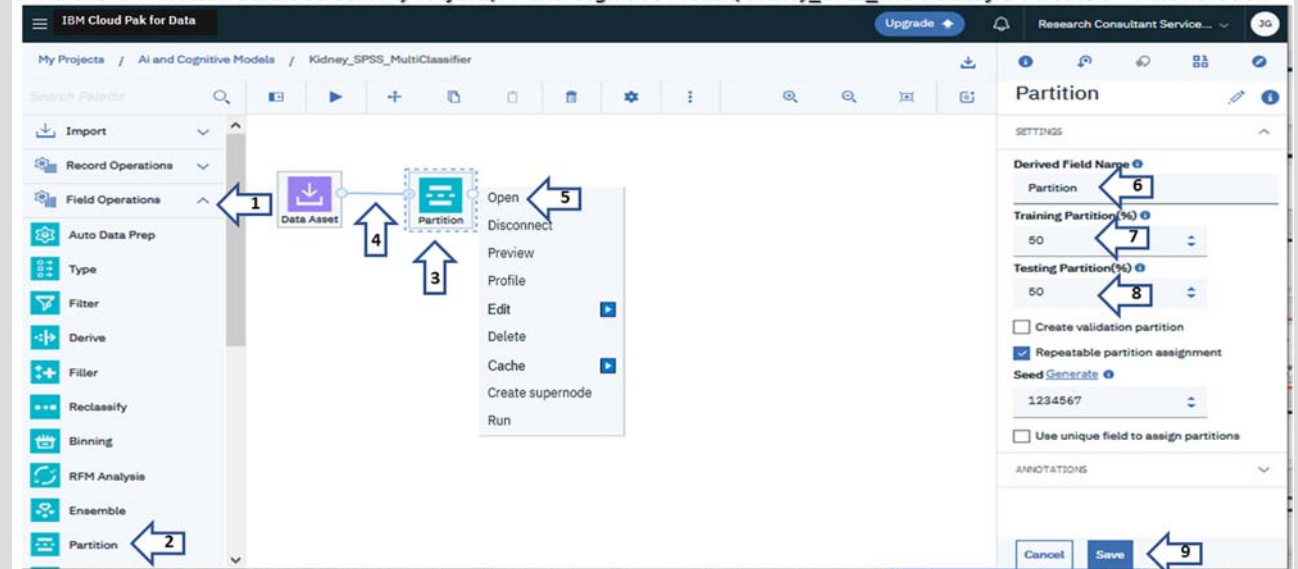
(Continued)

Slide Description

Screen figure

14 In the IBM Watson Studio screen: "My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier": create a "Data Partition"
Select "Field Operations" to expand its menu, click and drag "Partition" icon to the Modeler Flow, join the nodes, right click "Partition" and select "Open." In "Partition" configure "Derived Field Name" as "Partition," "Training Partition at 50%," "Testing Partition as 50%" and click "Save"

14. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier: create a "Data Partition"



Select "Field Operations" to expand its menu, click and drag "Partition" icon to the Modeler Flow, join the nodes, right click "Partition" and select "Open". In "Partition" configure "Derived Field Name" as "Partition," "Training Partition at 50%," "Testing Partition as 50%" and click "Save"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

15 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier*: Specify output "Type"
Select "Field Operations" to expand its menu, click and drag "Type" icon to the Modeler Flow, join the nodes with "Partition", select "Type" icon make a right click and select "Open." Select the option "Read metadata," click the button "Read Values," change in "Class" the role to "Target" and click "Save"

15. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier*: Specify output "Type"

Field	Measure	Role	Value mode	Values	Check
<input type="checkbox"/> htn	Flag	Input	Instantiated	no, yes	None
<input type="checkbox"/> dm	Flag	Input	Instantiated	no, yes	None
<input type="checkbox"/> cad	Flag	Input	Instantiated	no, yes	None
<input type="checkbox"/> appet	Nominal	Input	Instantiated	? good, poor	None
<input type="checkbox"/> pe	Nominal	Input	Instantiated	? no, yes	None
<input type="checkbox"/> ane	Nominal	Input	Instantiated	? no, yes	None
<input type="checkbox"/> class	Flag	Target	Instantiated	ckd, notckd	None
<input type="checkbox"/> Partition	Nominal	Non	Instantiated	1_Training_2_Test...	None

Select "Field Operations" to expand its menu, click and drag "Type" icon to the Modeler Flow, join the nodes with "Partition", select "Type" icon make a right click and select "Open". Select the option "Read metadata", click the button "Read Values", change in "Class" the role to "Target" and click "Save".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide **Description**

16 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier*: specify "Auto Classifiers"
Select "Modeling" to expand its menu, click and drag "Auto Classifiers" icon to the Modeler Flow, join the nodes with "Type," select "Class" icon make a right click and select "Open"
Note: SPSS automatically detect the field "class" as a target to classify, accept the default values clicking "Save." Finally, "Run" the model as shown in the slide

Screen figure

16. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Kidney_SPSS_MultiClassifier*: specify "Auto Classifiers"

The screenshot displays the IBM Watson Studio interface for a project named "Kidney_SPSS_MultiClassifier". The main workspace shows a modeler flow with four nodes: "Data Asset", "Partition", "Type", and "class". A context menu is open over the "class" node, listing options: "Open", "Disconnect", "Preview", "Profile", "Edit", "Delete", "Cache", "Create supernode", and "Run". The right-hand panel shows the "Inputs" section with "Field name" checked. A "Save" button is visible at the bottom right of the panel. Numbered arrows (1-8) indicate the sequence of actions described in the text.

Select "Modeling" to expand its menu, click and drag "Auto Classifiers" icon to the Modeler Flow, join the nodes with "Type" , select "Class" icon make a right click and select "Open". Note: SPSS automatically detect the field class as a target to classify, accept the default values clicking "Save". Finally, "Run" the model as shown in the slide.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

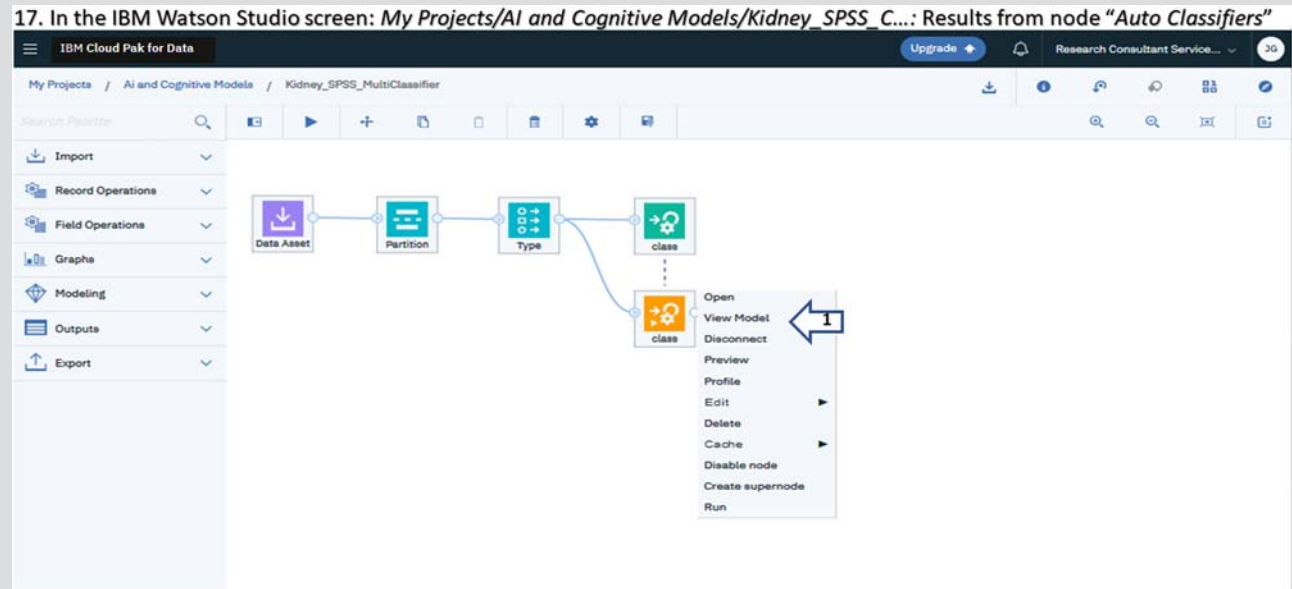
(Continued)

(Continued)

Slide **Description**

17 In the IBM Watson Studio screen: “My Projects/AI and Cognitive Models/Kidney_SPSS_Classifier”:
Results from node “Auto Classifiers”
Select the results from colored icon “Auto Classifiers” form, make a right click and select “View Model”

Screen figure



Select the results from yellow icon “Auto Classifiers” form , make a right click and select “View Model”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

18

In the IBM Watson Studio screen: "My Projects/AI and Cognitive Models/Kidney_SPSS_Classifier": Evaluate "Auto Classifiers Models" The top four "estimators" from "Auto Classifier Models" have very similar accuracy, click on the link for "Random Trees" to see details of its model evaluation "with Model Accuracy = 100," then check the "Linear SVM" with a "Model Accuracy = 99.454"; In the Auto Classifier screen de-select all the other estimator except "Linear SVM*" and go back to the Modeler flow selecting "Kidney_SPSS_MultiClassifier"

Note*: You can select any one of the models as the best

18. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Kidney_SPSS_C...: Evaluate "Auto Classifiers Models"

SEL	MODEL NAME	ESTIMATOR	BUILD TIME (MIN)	NO. FIELDS USED	ACCURACY	ACCUMULATED ACCURACY	AREA UNDER CURVE	ACCUMULATED AUC	ACTIONS
<input type="checkbox"/>	XGBoost Tree 1	XGBoost Tree 1	2	24	99.454	99.454	0.999	0.999	
<input type="checkbox"/>	XGBoost Linear 1	XGBoost Linear 1	2	24	99.381	99.381	0.999	0.999	
<input type="checkbox"/>	Random Trees 1	Random Trees 1	2	24	100.000	100.000	1.000	1.000	
<input checked="" type="checkbox"/>	LSVM 1	LSVM 1	2	24	99.454	99.454	1.000	1.000	

Model Evaluation Measures	
Accuracy	0.944
Weighted True Positive Rate	0.944
Weighted False Positive Rate	0.052
Weighted Precision	0.945
Weighted Recall	0.944
Weighted F ₁ Measure	0.944

Model Evaluation Measures	
Accuracy	1.000
Weighted True Positive Rate	1.000
Weighted False Positive Rate	0.000
Weighted Precision	1.000
Weighted Recall	1.000
Weighted F ₁ Measure	1.000

The top four "estimators" from "Auto Classifier Models" have very similar accuracy, click on the link for "Random Trees" to see details of its model evaluation with "Model Accuracy=0.944", then check for "Linear SVM" with "Model Accuracy=1.0"; In the Auto Classifier screen de-select all the other estimator and go back to the Modeler flow selecting "Kidney_SPSS_MultiClassifier"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

19 In the IBM Watson Studio screen: "My Projects/AI and Cognitive Models/Kidney_SPSS_Multiclassifier": Analysis for "Auto Classifier"
Kidney_SPSS_Multiclassifier": Analysis for "Auto Classifier"
Select "Outputs" to expand its menu, click and drag "Analysis" icon to the Modeler Flow, join the nodes with "Auto Classifier" yellow icon, select "Analysis" icon makes a right click and select "Open." Select the following for the settings: "Coincidence Matrices," "Performance evaluations," "Separate by partitions," click "Save." With right click on "Analysis" and "Run" the node as indicated in the slide

19. In IBM Watson Studio screen: My Projects/AI and Cognitive Models/Kidney_SPSS_Multiclassifier: Analysis for "Auto Classifier"

The screenshot displays the IBM Watson Studio interface for a project named "Kidney_SPSS_Multiclassifier". The main workspace shows a workflow with nodes: "Data Asset", "Partition", "Type", "class", and "Analysis". The "Analysis" node is highlighted, and a context menu is open over it, showing options: "Open", "Disconnect", "Preview", "Profile", "Edit", "Delete", "Cache", "Create supernode", and "Run". The "Analysis" settings panel is also visible, showing options for "Coincidence matrices", "Performance evaluation", and "Separate by partition". Numbered callouts 1-9 indicate specific actions: 1. Expand "Outputs" menu; 2. Select "Analysis" icon; 3. Drag "Analysis" icon to the workflow; 4. Connect nodes; 5. Right-click "Analysis" node; 6. Select "Open"; 7. Check "Coincidence matrices"; 8. Click "Save"; 9. Select "Run".

Select "Outputs" to expand its menu, click and drag "Analysis" icon to the Modeler Flow, join the nodes with "Auto Classifier" yellow icon, select "Analysis" icon makes a right click and select "Open". Select the following for the settings: "Coincidence Matrices", "Performance evaluations", "Separate by partitions", click "Save". With right click on "Analysis" and "Run" the node

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

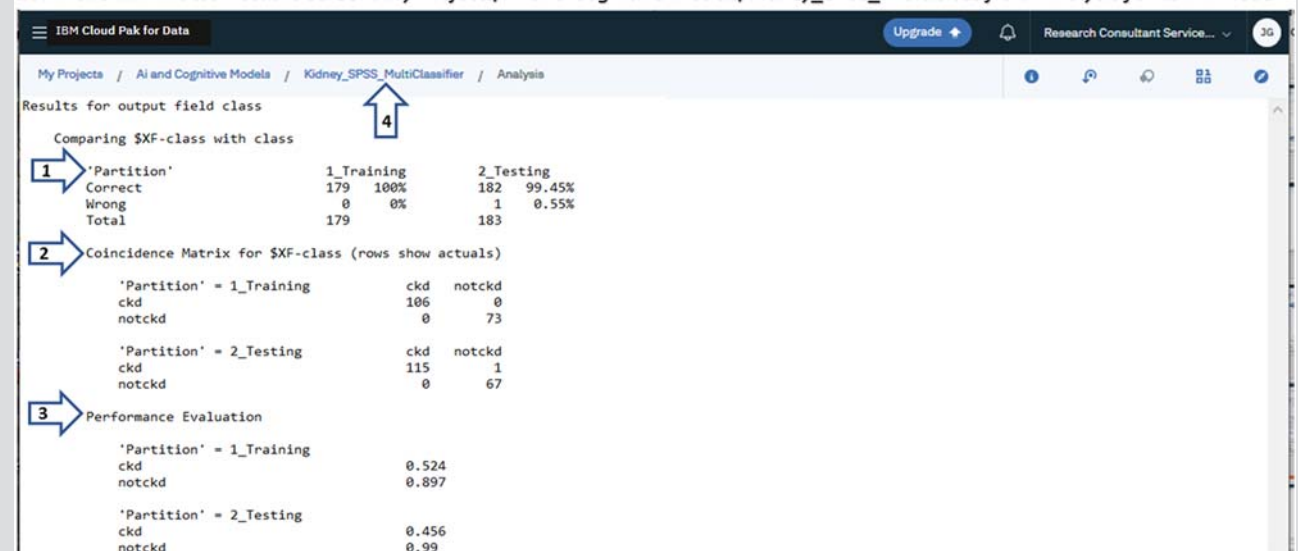
(Continued)

(Continued)

Slide 20 Description
In the IBM Watson Studio screen: "My Projects/AI and Cognitive Models/Kidney_SPSS_Multiclassifier": "Analysis for LSVM model"
The Analysis show that the "Linear SVM" models shows the 2 partitions for "class": "1_Training = 100%" and "2_Testing = 99.45%" (almost 100%), the "Coincide Matrix" and the "Performance Evaluation." Go back to the "Model Flow" clicking in: "Kidney_SPSS_MultiClassifier"

Screen figure

20. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Kidney_SPSS_Multiclassifier: "Analysis for LSVM model"



The Analysis show that the "Linear SVM" models shows the 2 partitions for "class": "1_Training=100%" and "2_Testing=99.45%" (almost 100%), the "Coincide Matrix" and the "Performance Evaluation". Go back to the "Model Flow" clicking in: "Kidney_SPSS_MultiClassifier"

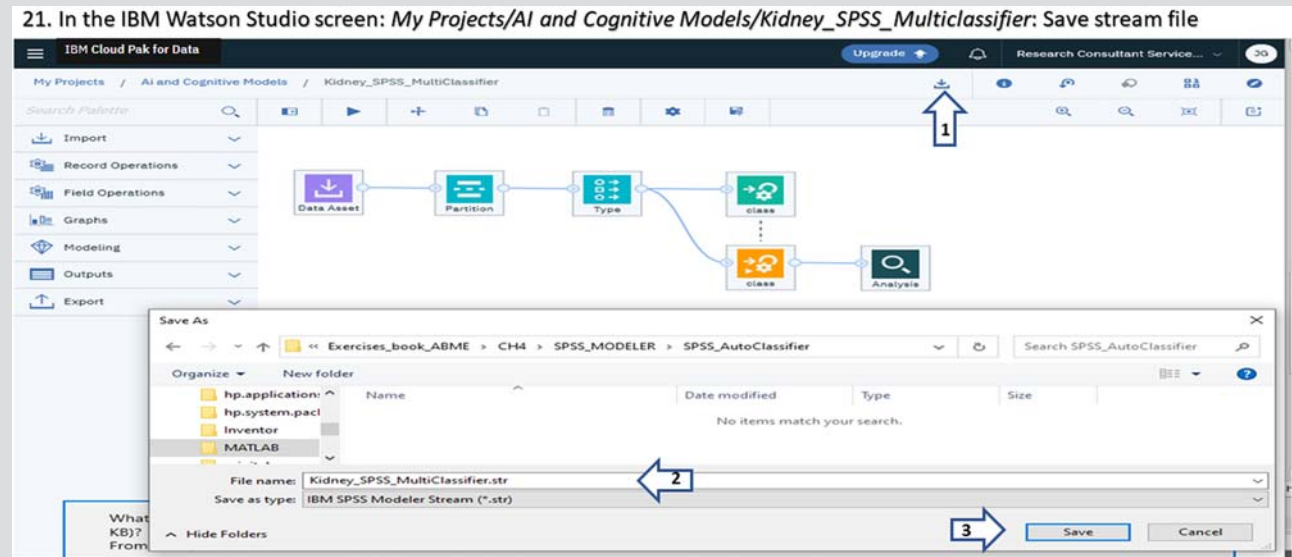
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 21 Description In the IBM Watson Studio screen: "My Projects/AI and Cognitive Models/Kidney_SPSS_Multiclassifier": "Save stream file" Save the streams file as an "STR file," click on "Download" icon at the top, go to the sub directory "Exercises_book_ABME\CH4 \SPSS_MODELER \SPSS_MultiClassifier," enter the name: "Kidney_SPSS_MultiClassifier.str" and click the button "Save." * It must be the scoring branch pre-configured for deployment (it is the path through the stream which will be executed when the deployment job is run)

Screen figure



Save the streams file as an "STR file", click on "Download" icon at the top, go to the sub directory "Exercises_book_ABME\CH4\SPSS_MODELER\SPSS_MultiClassifier", enter the name: "Kidney_SPSS_MultiClassifier.str" and click the button "Save". * It must be the scoring branch pre-configured for deployment (it is the path through the stream which will be executed when the deployment job is run)

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

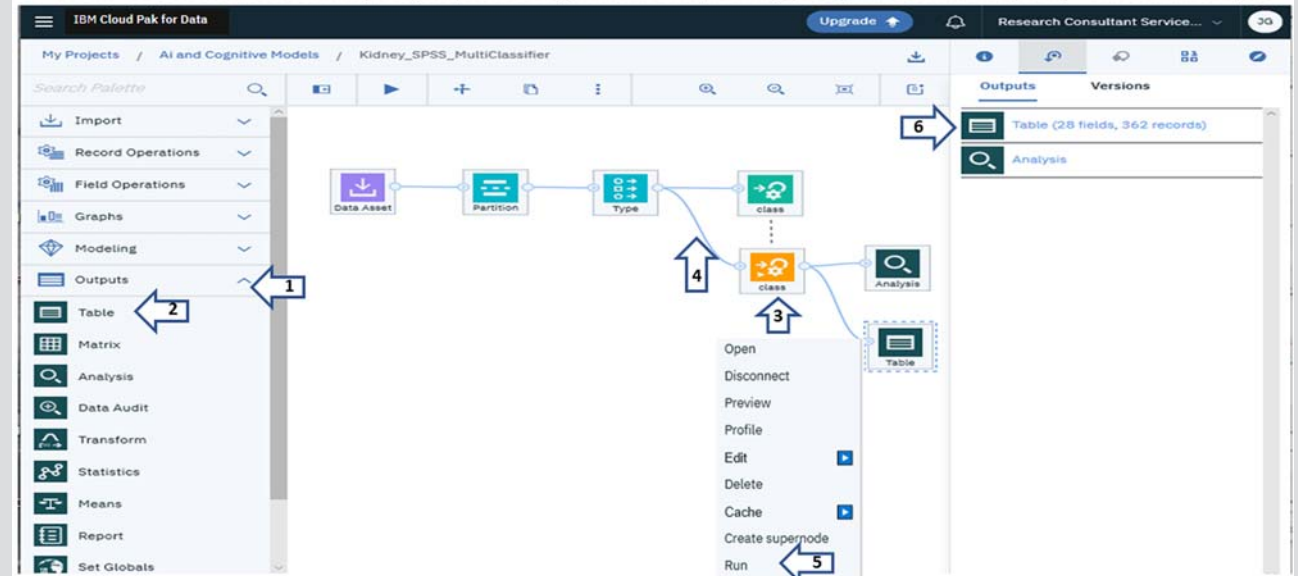
(Continued)

Slide **Description**

22 In IBM Watson Studio screen: My Projects/AI and Cognitive Models/Kidney_SPSS_MutiClassifier: Specify "Table as an output"
Select "Outputs" to expand its menu, click and drag "Table" icon to the Modeler Flow, join the nodes with "Auto Classifier" colored icon, select "Table" icon, make a right click and select "Run." In the "Outputs Tab" click on the results for the table

Screen figure

22. In IBM Watson Studio screen: My Projects/AI and Cognitive Models/Kidney_SPSS_MutiClassifier: Specify "Table as an output"



Select "Outputs" to expand its menu, click and drag "Table" icon to the Modeler Flow, join the nodes with "Auto Classifier" colored icon, select "Table" icon, make a right click and select "Run". In the "Outputs Tab" click on the results for the table.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 23 Description In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Kidney_SPSS_M...: See results "Table as an output" Table screen: see results stored in table, the last two columns contains the prediction: "\$XF-class" has "prediction of kidney disease" and "\$XFC-class" has the "confidence score for the prediction." Return to the project with a click "AI and Cognitive Models"

Screen figure

23. In the IBM Watson Studio screen: My Projects/AI and Cognitive Models/Kidney_SPSS_M...: See results "Table as an output"

age	bp	sg	al	rbc	pc	pcc	ba	bgr	bu	sc	sod	pot	hemo	pcv	wbcc	rbcc	htn	dm	cad	appet	pe	ane	class	Partition	\$XF-class	\$XFC-class	
48	80	1.02	1	0	?	normal	notpresent	notpresent	121	36	1.200	?	?	15.4	44	7800	5.2	yes	yes	no	good	no	no	ckd	1_Training	ckd	0.866
7	50	1.02	4	0	?	normal	notpresent	notpresent	?	18	0.800	?	?	11.3	38	6000	?	no	no	no	good	no	no	ckd	1_Training	ckd	0.882
62	80	1.01	2	3	normal	normal	notpresent	notpresent	423	53	1.800	?	?	9.6	31	7500	?	no	yes	no	poor	no	yes	ckd	2_Testing	ckd	0.939
48	70	1.005	4	0	normal	abnormal	present	notpresent	117	56	3.800	111	2.5	11.2	32	6700	3.9	yes	no	no	poor	yes	yes	ckd	2_Testing	ckd	0.957
51	80	1.01	2	0	normal	normal	notpresent	notpresent	106	26	1.400	?	?	11.6	35	7300	4.6	no	no	no	good	no	no	ckd	1_Training	ckd	0.911
60	90	1.015	3	0	?	?	notpresent	notpresent	74	25	1.100	142	3.2	12.2	39	7800	4.4	yes	yes	no	good	yes	no	ckd	1_Training	ckd	0.775
68	70	1.01	0	0	?	normal	notpresent	notpresent	100	54	24.000	104	4	12.4	36	?	?	no	no	no	good	no	no	ckd	2_Testing	ckd	0.979
52	100	1.015	3	0	normal	abnormal	present	notpresent	138	60	1.900	?	?	10.8	33	9600	4	yes	yes	no	good	no	yes	ckd	1_Training	ckd	0.951
53	90	1.02	2	0	abnormal	abnormal	present	notpresent	70	107	7.200	114	3.7	9.5	29	12100	3.7	yes	yes	no	poor	no	yes	ckd	1_Training	ckd	0.977
50	60	1.01	2	4	?	abnormal	present	notpresent	490	55	4.000	?	?	9.4	28	?	?	yes	yes	no	good	no	yes	ckd	1_Training	ckd	0.976
63	70	1.01	3	0	abnormal	abnormal	present	notpresent	380	60	2.700	131	4.2	10.8	32	4500	3.8	yes	yes	no	poor	yes	no	ckd	2_Testing	ckd	0.968
68	70	1.015	3	1	?	normal	present	notpresent	208	72	2.100	138	5.8	9.7	28	12200	3.4	yes	yes	yes	poor	yes	no	ckd	2_Testing	ckd	0.967
68	70	?	?	?	?	?	notpresent	notpresent	98	86	4.600	135	3.4	9.8	?	?	?	yes	yes	yes	poor	yes	no	ckd	2_Testing	ckd	0.964
68	80	1.01	3	2	normal	abnormal	present	present	157	90	4.100	130	6.4	5.6	16	11000	2.6	yes	yes	yes	poor	yes	no	ckd	1_Training	ckd	0.966

Table screen: see results stored in table, the last two columns contains the prediction: "\$XF-class" has "prediction of kidney disease" and "\$XFC-class" has the "confidence score for the prediction". Return to the project with a click "AI and Cognitive Models"

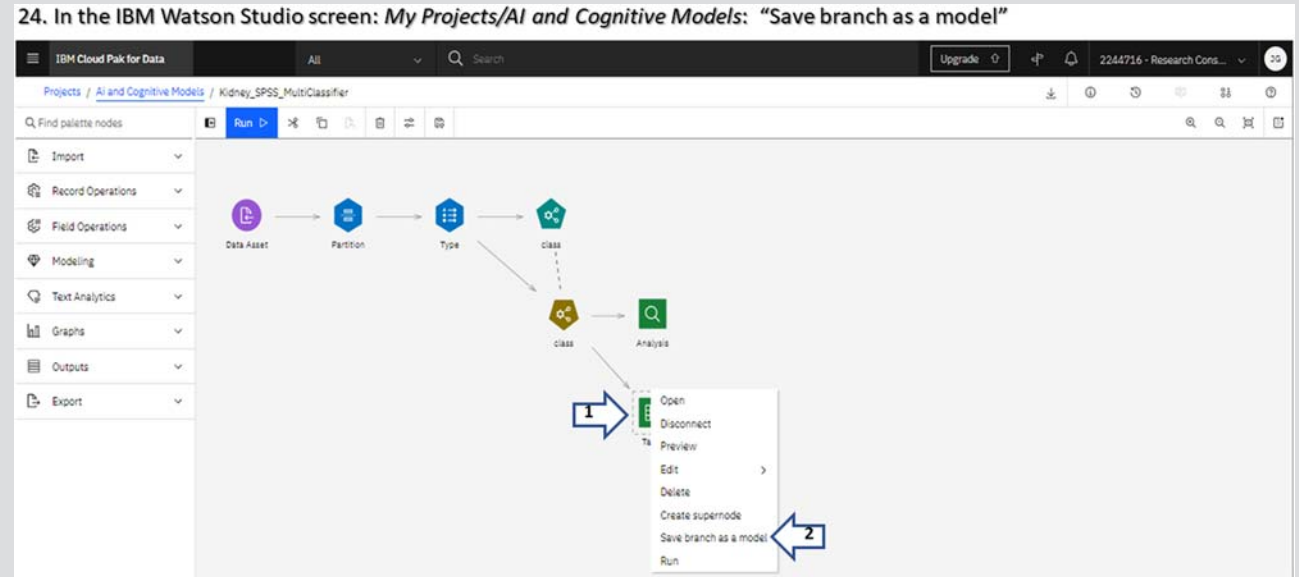
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 24 In the IBM Watson Studio screen: My Projects/AI and Cognitive Models: "Save branch as a model" Select "Table" and with right click "Save branch as a model" as shown in the slide

Screen figure



Select "Table" and with right click "Save branch as a model" as shown in the slide

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

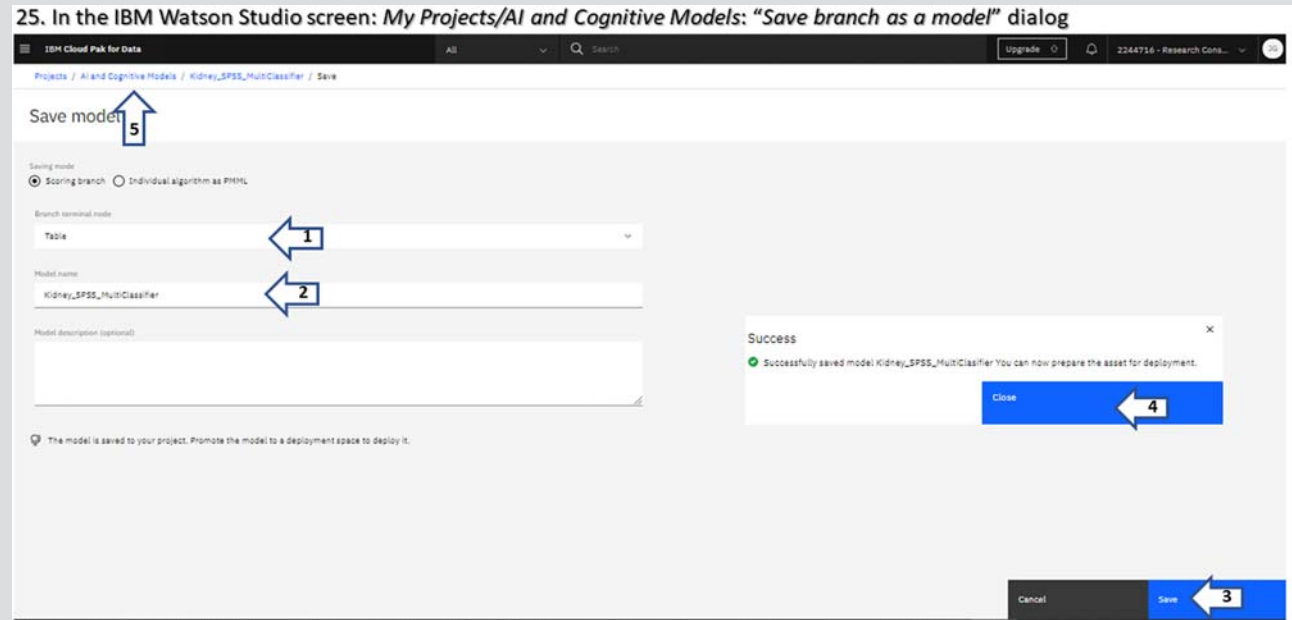
(Continued)

(Continued)

Slide **Description**

25 In the IBM Watson Studio screen: My Projects/AI and Cognitive Models: "Save branch as a model" dialog
In the "Save model" dialog: select mode as "Scoring branch," branch terminal mode as "Table," model name as "Kidney_SPSS_MultiClassifier," select the lower "Save" button. After the "Success" message click "Close" and go back to the "AI and Cognitive Models"

Screen figure



In the "Save model" dialog: select mode as "Scoring branch", branch terminal mode as "Table", model name as "Kidney_SPSS_MultiClassifier", select the lower "Save" button. After the "Success" message click "Close" and go back to the "AI and Cognitive Models"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

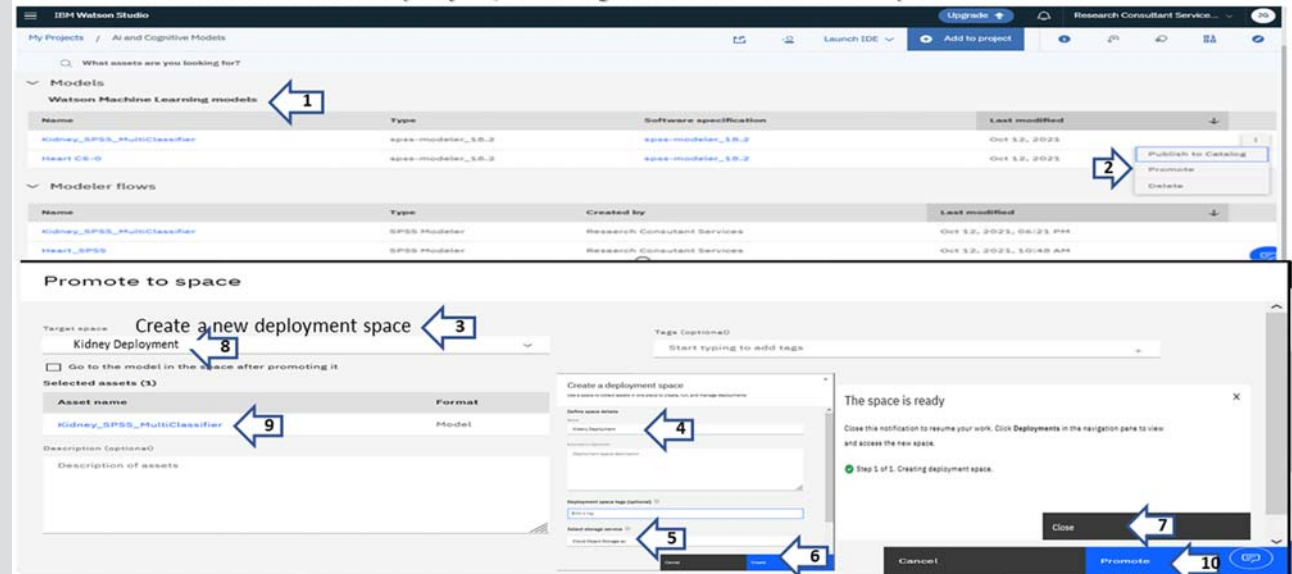
(Continued)

Slide Description

Screen figure

26 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models: "Data assets"* promote the dataset
In the upper screen "*My projects/ AI and Cognitive Models*" in the section "*Models Watson Machine Learning*" expand menu of "*Kidney_SPSS_MultiClassifier*", click on "*Promote*". In the lower screen Windows "*Create a deployment space*" enter name = "*Kidney Deployment*", select storage service = "*your assigned storage*" click on "*Create*", and when space is ready click "*Close*". At Promote to space define Target space = "*Kidney Deployment*", Data assets = "*Kidney_diaseses.csv*" and finally "*Promote to space*"

26. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models: "Data assets"* promote the dataset



In the upper screen "*My projects/ AI and Cognitive Models*" in the section "*Models Watson Machine Learning*" expand menu of "*Kidney_SPSS_MultiClassifier*", click on "*Promote*". In the lower screen Windows "*Create a deployment space*" enter name = "*Kidney Deployment*", select storage service = "*your assigned storage*" click on "*Create*", and when space is ready click "*Close*". At Promote to space define Target space = "*Kidney Deployment*", Data assets = "*Kidney_diaseses.csv*" and finally "*Promote to space*"

From BOOK APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description Screen figure

27 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models: "Our project status"* and going to *"Spaces"* Verify your project actual components: *"Data assets are 5"*, *"Models Watson Machine Learning 2"*, and *"Modeler flows are 3"*. Click on the *"Navigation menu"*, select *"Deployment"* and *"View all spaces"* as indicated.

27. In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models: "Our project status"* and going to *"Spaces"*

The screenshot displays the IBM Watson Studio interface for a project named "AI and Cognitive Models". On the left is a navigation menu with options: Home, Data, Projects, Deployments, View all spaces, No recent spaces, Services, Gallery, Administration, and Support. The main content area is divided into three sections:

- Data assets:** A table with 5 rows. Columns: Name, Type, Created by, Last modified. Assets include kidney_disease.csv, heart_disease.csv, cluster1.csv, cluster2.csv, and diabetes.csv.
- Models:** A section titled "Watson Machine Learning models" with a table of 2 rows. Columns: Name, Type, Software specification, Last modified. Models include kidney_SPSS_MultiClassifier and heart_CS-O.
- Modeler flows:** A table with 3 rows. Columns: Name, Type, Created by, Last modified. Flows include kidney_SPSS_MultiClassifier, heart_SPSS, and diabetes_SPSS_1.

Verify your project actual components: *"Data assets are 5"*, *"Models Watson Machine Learning 2"*, and *"Modeler flows are 3"* as shown. Click on the *"Navigation menu"*, select *"Deployment"* and *"View all spaces"* as indicated

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

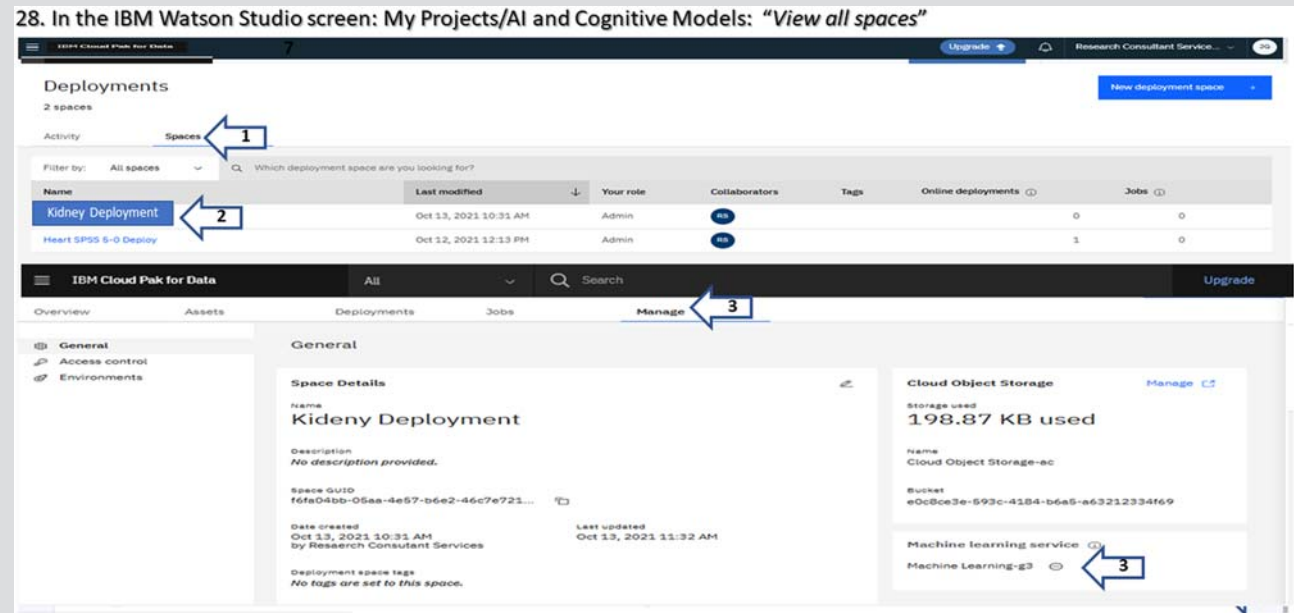
(Continued)

(Continued)

Slide Description

Screen figure

28 In the IBM Watson Studio screen: My Projects/AI and Cognitive Models: "View all spaces"
In upper screen is shown "Deployments" click on "Spaces" and "Kidney Deployment". Then as shown in lower screen click on "Manage" and associate "your assigned Machine Learning service"



In upper screen is shown "Deployments" click on "Spaces" and "Kidney Deployment". Then as shown in lower screen click on "Manage" and associate "your assigned Machine Learning service"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

29 In the IBM Watson Studio screen: "All Deployment spaces"
On the section "Assets" on "Kidney_SPSS_Multiclassifier" select "Deploy" as indicated at the upper screen. At lower screen on windows "Deploy" indicates deployment type "Online", name = "Kidney_SPSS_MultiClassifier", service name = "kidney_deploy" and click "Create". Note: Probably a error message could be shown as: This deployment cannot be processed because it exceeds the allocated capacity unit hours (CUH). Increase the compute resources for this job and try again. The free version "Lite" has a capacity Unit Hours included of 20 por month. and we have to upgrade "standard" billed per CUH or professional with 2500 per month.

Screen figure

29. In the IBM Watson Studio screen: "All Deployment spaces"

The screenshot displays the IBM Watson Studio interface for the 'Kidney_SPSS_MultiClassifier' asset. The 'Assets' tab is selected, and a table lists the model 'Kidney_SPSS_MultiClassifier'. A 'Deploy' button is visible next to the model. Below the table, the 'Create a deployment' form is shown with the 'Online' deployment type selected. The form includes fields for name, serving name, and software specification, along with a 'Create' button.

On the section "Assets" on "Kidney_SPSS_Multiclassifier" select "Deploy" as indicated at the upper screen. At lower screen on windows "Deploy" indicates deployment type "Online", name = "Kidney_SPSS_MultiClassifier", service name = "kidney_deploy" and click "Create"

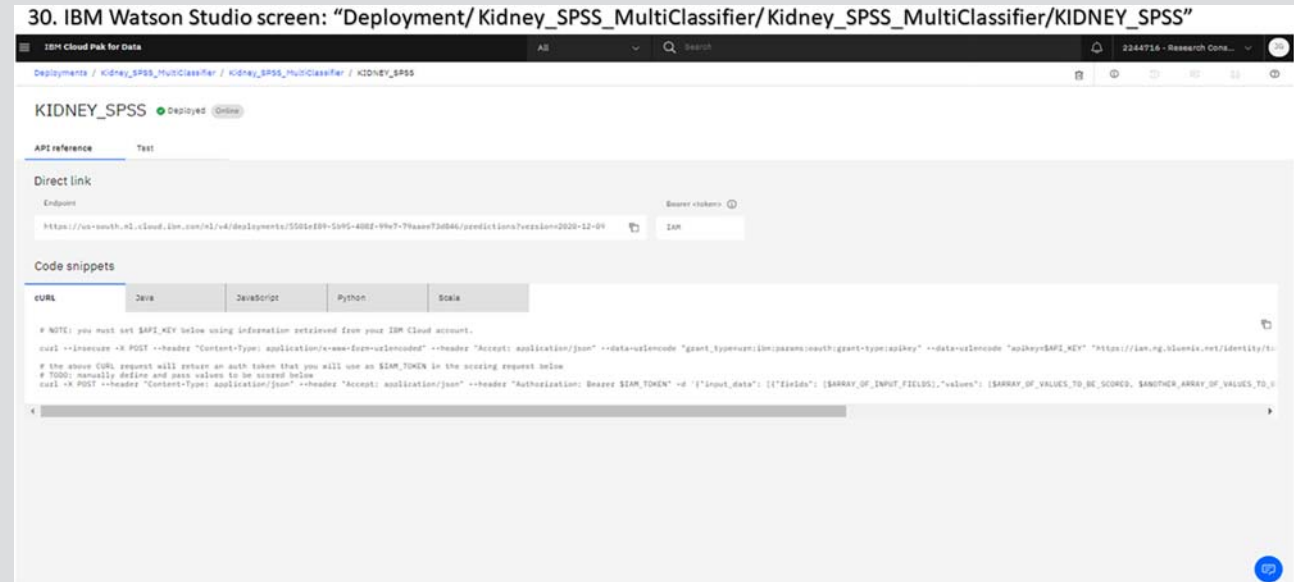
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 30
Description
IBM Watson Studio screen:
"Deployment/
Kidney_SPSS_MultiClassifier/
Kidney_SPSS_MultiClassifier/
KIDNEY_SPSS"
Copy the "API reference > Direct
Link > Endpoint" in notepad and
save it as "...\Exercises_book_ABME
\CH4\SPSS_MODELER
\SPSS_AutoClassifiers
\IBM_Cloud_Shell.txt" for later use.
Observe the "Code Snippets"
for: "cURL," "Java," "JavaScript,"
"Python," and "Scala." Finally select
the "Test" tab

Screen figure



Copy the "API reference > Direct Link > Endpoint" in notepad and save it as "...\Exercises_book_ABME\CH4\SPSS_MODELER\SPSS_AutoClassifiers\IBM_Cloud_Shell.txt" for later use. Observe the "Code Snippets" for: "cURL", "Java", "JavaScript", "Python" and "Scala". Finally select the "Test" tab.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

31 IBM Watson Studio screen: "Deployment/ Kidney_SPSS_MultiClassifier/ Kidney_SPSS_MultiClassifier/ KIDNEY_SPSS" Test
For test the model in the tab "Test" enter the following data: in "AGE 62," "BP 80," "SG 1.01," "AL 2," "SU 3," "RBC normal," "PC normal," "PCC notpresent," "BA 423," "BGR 53," "BU 1.8," "SC 0.8," "SOD 111," "POT 2.5," "HEMO 9.6," "PCV 31," "WBCC 7500," "RBCC 3.9," "HTN no," "DM yes," "CAD no," "APPET poor," "PE no," "ANE yes," "Class left empty space." Press the button "Predict" and the answer is indicated in "JSON" that predict value "CLASS = notckd" with "CONFIDENCE = 0.608". Finally, "Log out" as shown

31. IBM Watson Studio screen: "Deployment/ Kidney_SPSS_MultiClassifier/ Kidney_SPSS_MultiClassifier/KIDNEY_SPSS" Test

age	bp	sg	al	su	rbc	pc	ba	bgr	bu	sc	sod	pot	hemo	pcv	wbcc	rbcc	htn	dm	cad	appet	pe	ane	class
62	80	1.01	2	3	normal	normal	423	53	1.8	0.8	111	2.5	9.6	31	7500	3.9	no	yes	no	poor	no	yes	


```
{
  "CLASS": "notckd",
  "CONFIDENCE": 0.608
}
```

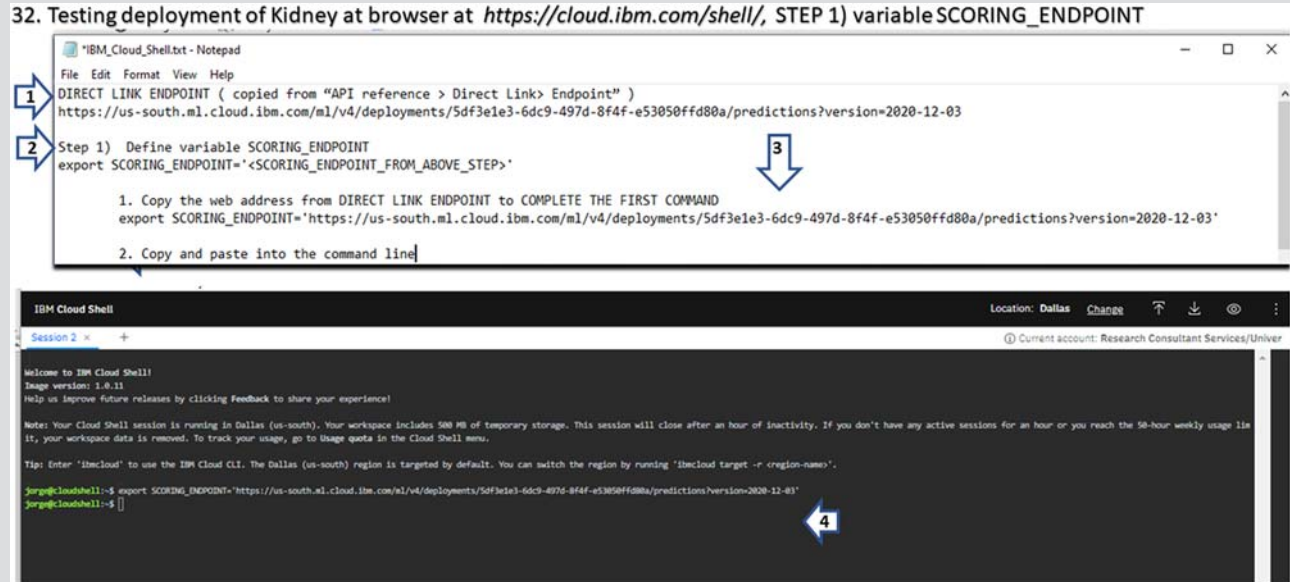
For test the model in the tab "Test" enter the following data: in "AGE 62," "BP 80," "SG 1.01," "AL 2," "SU 3," "RBC normal," "PC normal," "PCC notpresent," "BA 423," "BGR 53," "BU 1.8," "SC 0.8," "SOD 111," "POT 2.5," "HEMO 9.6," "PCV 31," "WBCC 7500," "RBCC 3.9," "HTN no," "DM yes," "CAD no," "APPET poor," "PE no," "ANE yes," "Class left empty space". Press the button "Predict" and the answer is indicated in "JSON" that predict value "CLASS=notckd" with "CONFIDENCE=0.608". Finally, "Log out" as shown.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza UIloa

(Continued)

(Continued)

Slide	Description	Screen figure
32	Testing deployment of Kidney in browser at https://cloud.ibm.com/shell/ , Step 1) variable SCORING_ENDPOINT Follow the instruction to complete the command line to define the variable "SCOREPOINT" as indicated in the top screen. When ready, copy and paste it as shown in the lower screen	 <p>32. Testing deployment of Kidney at browser at https://cloud.ibm.com/shell/, STEP 1) variable SCORING_ENDPOINT</p> <p>1. DIRECT LINK ENDPOINT (copied from "API reference > Direct Link> Endpoint") <code>https://us-south.m1.cloud.ibm.com/ml/v4/deployments/5df3e1e3-6dc9-497d-8f4f-e53050ffd80a/predictions?version=2020-12-03</code></p> <p>2. Step 1) Define variable SCORING_ENDPOINT <code>export SCORING_ENDPOINT='<SCORING_ENDPOINT_FROM_ABOVE_STEP>'</code></p> <p>3. 1. Copy the web address from DIRECT LINK ENDPOINT to COMPLETE THE FIRST COMMAND <code>export SCORING_ENDPOINT="https://us-south.m1.cloud.ibm.com/ml/v4/deployments/5df3e1e3-6dc9-497d-8f4f-e53050ffd80a/predictions?version=2020-12-03"</code> 2. Copy and paste into the command line</p> <p>4. <code>jorga@cloudshell:~\$ export SCORING_ENDPOINT="https://us-south.m1.cloud.ibm.com/ml/v4/deployments/5df3e1e3-6dc9-497d-8f4f-e53050ffd80a/predictions?version=2020-12-03"</code> <code>jorga@cloudshell:~\$</code></p>



Follow the instruction to complete the command line to define the variable "SCOREPOINT" as indicated in the top screen. When ready, copy and paste it as shown in the lower screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 33 Description Testing deployment of Kidney in browser <https://cloud.ibm.com/shell/>, Step 2) Generate Token Step 2) Generate Token: “*ibmcloud* command” indicate in the upper screen, and paste it as shown in the lower screen, to obtain the TOKEN as shown in the lower screen, this will be used in the next step

Screen figure

33. Testing deployment of Kidney in browser at <https://cloud.ibm.com/shell/>, STEP 2) Generate Token



1 Step 2) Second COMMAND "Generate a TOKEN"
`ibmcloud iam oauth-tokens --output JSON | jq -r .iam_token`
1.- Copy and paste into the command line "AS A PLAIN TEXT" to obtain a TOKEN

2

STEP 2) Generate Token: “*ibmcloud* command” indicate in the upper screen, and paste it as shown in the lower screen, to obtain the TOKEN as shown in the lower screen, this will be used in the next step

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 34 Description Screen figure

34 IBM Cloud Shell at <https://cloud.ibm.com/shell>, Step 3) Define variable IAM_TOKEN: Copy the TOKEN generated in the last slide, and paste in the command as shown in the upper screen, then copy the entire command into the IBM Cloud Shell as shown in the lower screen

34. IBM Cloud Shell at <https://cloud.ibm.com/shell> Step 3) Define variable IAM_TOKEN

The image shows two screenshots. The top screenshot is a Notepad window titled "IBM_Cloud_Shell.txt" with the following content:

```
Step 3) Define Variable IAM_TOKEN
export IAM_TOKEN="<IAM_TOKEN>"

1.- Copy the TOKEN generated and Replace it in the variable
export IAM_TOKEN="ldDIi1cmVhbG:
Dgjc58ppERpIu2RUHZtEUG75G_v97p3jtEMn3NpIET1HTsk1hGaAcctJ-FHU_NgTam5f0g8mDt9RGyxrVcSRUogCjBwe6sZzzR6YzLNI4ddBCQ"

2. Copy and paste into the command line
```

The bottom screenshot is the IBM Cloud Shell terminal. It shows the command being executed:

```
jorga@cloudshell:~$ export SCOPING_ENDPOINT="https://us-south.e1.cloud.ibm.com/v1/deployments/5df3ae3-6dc3-4874-484f-453059ff688a/predictions?version=2020-12-01"
jorga@cloudshell:~$ ibmcloud iam oauth-tokens --output JSON | jq -r '.iam_token'
{"token": "ldDIi1cmVhbG:
Dgjc58ppERpIu2RUHZtEUG75G_v97p3jtEMn3NpIET1HTsk1hGaAcctJ-FHU_NgTam5f0g8mDt9RGyxrVcSRUogCjBwe6sZzzR6YzLNI4ddBCQ"}
jorga@cloudshell:~$ export IAM_TOKEN="ldDIi1cmVhbG:
Dgjc58ppERpIu2RUHZtEUG75G_v97p3jtEMn3NpIET1HTsk1hGaAcctJ-FHU_NgTam5f0g8mDt9RGyxrVcSRUogCjBwe6sZzzR6YzLNI4ddBCQ"
jorga@cloudshell:~$
```

Step 3) Define variable IAM_TOKEN : Copy the TOKEN generated in the last slide, and paste in the command as shown in the upper screen, then copy the entire command into the IBM Cloud Shell as shown in the lower screen

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

35 IBM Cloud Shell at <https://cloud.ibm.com/shell>, Step 4) Define DATA and Obtain API response
Step 4) Define DATA and Obtain API response: Copy the Command with the real input data copy and paste in the command as shown in the lower screen. Note the values of the predicted result using this model.

35. IBM Cloud Shell at <https://cloud.ibm.com/shell> Step 4) Define DATA and Obtain API response

```
IBM_Cloud_Shell_Kidney.txt - Notepad
File Edit Format View Help
Step 4) Define data and obtain API response
curl -X POST \
  -H "Content-Type: application/json" \
  -H "Accept: application/json" \
  -H "Authorization: Bearer [token]" \
  -d '{"input_data": [{"input": [{"age": "65", "sex": "M", "height": "170", "weight": "70", "bmi": "24.2", "systolic_bp": "130", "diastolic_bp": "80", "sugar": "100", "cholesterol": "200"}]}]}' \
  https://api.ibm.com/v1/analyze

IBM Cloud Shell
Location: Dallas Change
Current account: 2244716 - Research Consultant Services/U
Session 1 x +
jorge@cloudshell:~$ curl -X POST \
  -H "Content-Type: application/json" \
  -H "Accept: application/json" \
  -H "Authorization: Bearer [token]" \
  -d '{"input_data": [{"input": [{"age": "65", "sex": "M", "height": "170", "weight": "70", "bmi": "24.2", "systolic_bp": "130", "diastolic_bp": "80", "sugar": "100", "cholesterol": "200"}]}]}' \
  https://api.ibm.com/v1/analyze
SCORING_ENDPOINT
```

3 Results

Step 4) Define DATA and Obtain API response: Copy the Command with the real input data and paste in the command as shown in the lower screen. Note the values of the predicted result using this model.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

We can have the following conclusions for “obtaining ML models using Auto Classifiers algorithms available at IBM Watson SPSS Modeler Flow for a kidney diseases dataset, and deploying the best model.”

- It was possible to predict when the patient has “Chronic kidney disease” based on the attributes “class” of the given dataset.
- It could be made with four “ML models” they are: “XGBoost-Linear,” “LinearSVM,” “XGBoost-Tree,” and “Random Trees.” These four “ML models” presented similar: “model evaluation accuracy,” “coincide matrices,” and “performance evaluation.”
- The deployment can be made in three easy steps:
 - Save the ML model stream to be used when the deployment job is run;
 - Save branch as a model directly from the “Auto Classifier” icon on SPSS Flow Modeler; and
 - Create deployment and test it in a “JASON” form as shown at slide 31.

Recommendations

- A larger dataset will help to analyze with more precision the “class” with more variations on the predicted value.
- Incorporate within the dataset, or create a new dataset that includes, new attributes that will help in the prediction, such as: “Smoking,” “Body Mass Index,” “Race,” “Family history of kidney disease,” “Abnormal kidney structure,” and others.

4.12.2.4 Research tutorial 4.4 IBM Watson AutoAI experimenter for “Breast cancer ML model and deploy the best model”

4.12.2.4.1 Case for research

“Obtain ML models using IBM Watson Auto AI experimenter for Breast Cancer dataset, and deploy the best model.”

4.12.2.4.2 General objective

“Use IBM Watson Studio AutoAI experimenter” to obtain ML for a Breast Cancer dataset and deploy the best model.

4.12.2.4.3 Background for “Breast cancer”

“Breast cancer” develops when cells in the breast mutate and grow out of control, forming a “breast tumor.” About 80% of them are “ductal carcinomas,” that begin in breast milk ducts, and 10% of them are “lobular carcinomas” that develop in the breast lobes or glands that produce milk [25]. “Breast cancer cells” can spread into the “lymph nodes” in and around the breasts and, from there, travel and form “tumors in distant parts of the body such as brain, bones, liver and lungs; this is identified as

“metastatic breast cancer.” “Breast cancer” also occurs in men, “Male breast cancer” accounts for 1% of all breast cancer diagnoses.

Factors that may increase the risk of “breast tumors” include:

- “Obesity,”
- “Breast density,”
- “Menstrual history,”
- “Lack of exercise”
- “Alcoholic drink,”
- “Previous medical treatment,”
- “Older age,” and
- “Gender.”

The most frequent “breast cancer types” are:

- “Ductal carcinoma”
- “Lobular carcinoma”
- “Adenocystic carcinoma”
- “Angiosarcoma”
- “Inflammatory breast cancer”
- “Metaplastic carcinoma”
- “Phyllodes tumor”

“Breast cancer subtypes” include those driven by specific “hormones” such as “estrogen,” “progesterone,” or the “protein HER2”:

- 60% of breast cancers are “estrogen-positive.”
- 20% of breast cancers are “HER2-positive”
- 20% are “triple-negative breast cancers,” which are among the more aggressive forms of the disease, they “test negative for estrogen, progesterone, and HER2.”

“Recurrent breast cancer” occurs when the disease returns after initial treatment. Most recurrent cancers appear within the first 2 or 3 years after treatment, but in some cases the cancer may recur many years later. Women with early breast cancer most often develop local recurrence within the first 5 years after treatment. On average, 7%–11% of women with early breast cancer experience a local recurrence during this time. There are three types of recurrent breast: “local recurrence,” “regional recurrence,” and “distant recurrence” [26]:

- “Local recurrence” is when the cancer has returned to the same location as the original cancer.
- “Regional recurrence” is when the cancer is found in or near the original location.
- “Distant recurrence” is when the breast cancer has spread to other parts of the body. This is also considered “metastatic breast cancer.”

4.12.2.4.4 Specific objectives

- Use “IBM Watson Auto AI experimenter for a Breast-cancer” dataset specific objectives are:
- Predict when patient has “Breast cancer” based on the attributes of a given dataset.

TABLE 4.3 Dataset “Breast-cancer.csv” fields and descriptions.

Field	Description missing attribute values are denoted by “?”
10 attributes	*286 instances of patients
age	Age in range years (nominal) = [10–19, 20–29, 30–39, 40–49, 50–59, 60–69, 70–79, 80–89, 90–99]
menopause	Menopause (nominal) = [‘lt40’, ‘ge40’, ‘premeno’] Where: <i>premeno</i> = premature menopause, lt40 = menopause less than 40 years old, ge40 = menopause greater equal than 40 years old
tumor-size	Tumor size (nominal) = [0–4, 5–9, 10–14, 15–19, 20–24, 25–29, 30–34, 35–39, 40–44, 45–49, 50–54, 55–59]
inv-nodes	<i>inv-nodes</i> = ‘lymph nodes’ (nominal) = [0–2, 3–5, 6–8, 9–11, 12–14, 15–17, 18–20, 21–23, 24–26, 27–29, 30–32, 33–35, 36–39]
node-caps	<i>node-caps</i> (nominal) = [‘yes’, ‘no’] * A metastasis may be contained within the lymph node with an intact node lymph capsule.
deg-malig	<i>deg-malig</i> (nominal) = [1,2,3] *Degree of malignancy, as estimated by loss of differentiation and increase of reproductive characteristics.
breast	<i>breast</i> (nominal) = [‘left’, ‘right’]
breast-quad	<i>breast-quad</i> (nominal) = [‘left-up’, ‘left-low’, ‘right-up’, ‘right-low’, ‘central’]
irradiat	<i>irradiat</i> : (nominal) = [‘yes’, ‘no’]
class	<i>class</i> (nominal) = [‘no-recurrence-events’, ‘recurrence-events no’]

Note: This dataset is available in the companion directory of the book, in the following directory: “. . . \MATLAB\Exercises_book_ABME\CH4\Watson_ML\Breast-cancer.csv”.

- Use “IBM Watson AutoAI” to simplify AI life cycle management by automating the following tasks:
 - Data preparation
 - Model development
 - Feature engineering
 - Hyperparameter optimization.
- Evaluate the seven best “ML models” for the “IBM Watson AutoAI experimenter.”
- Choose the best “ML model” pipeline based on “ROC AUC” optimized parameter.
- Save the best ML found.
- Create deployment and test it on input “JASON” string in a form.

4.12.2.4.5 Dataset

The dataset “Breast-cancer.csv” is a breast cancer domain obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. Breast cancer domain was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Slovenia. Thanks go to M. Zwitter and M. Soklic for providing the data, it is available at the UCI Machine Learning Repository [27].

This database contains nine attributes, from 286 instances (patients). The goal is “class” attribute, which refers to the tumor as “no-recurrence-events” and “recurrence-events.” This dataset, the descriptions, and values are indicated in Table 4.3.

4.12.2.4.6 Procedure

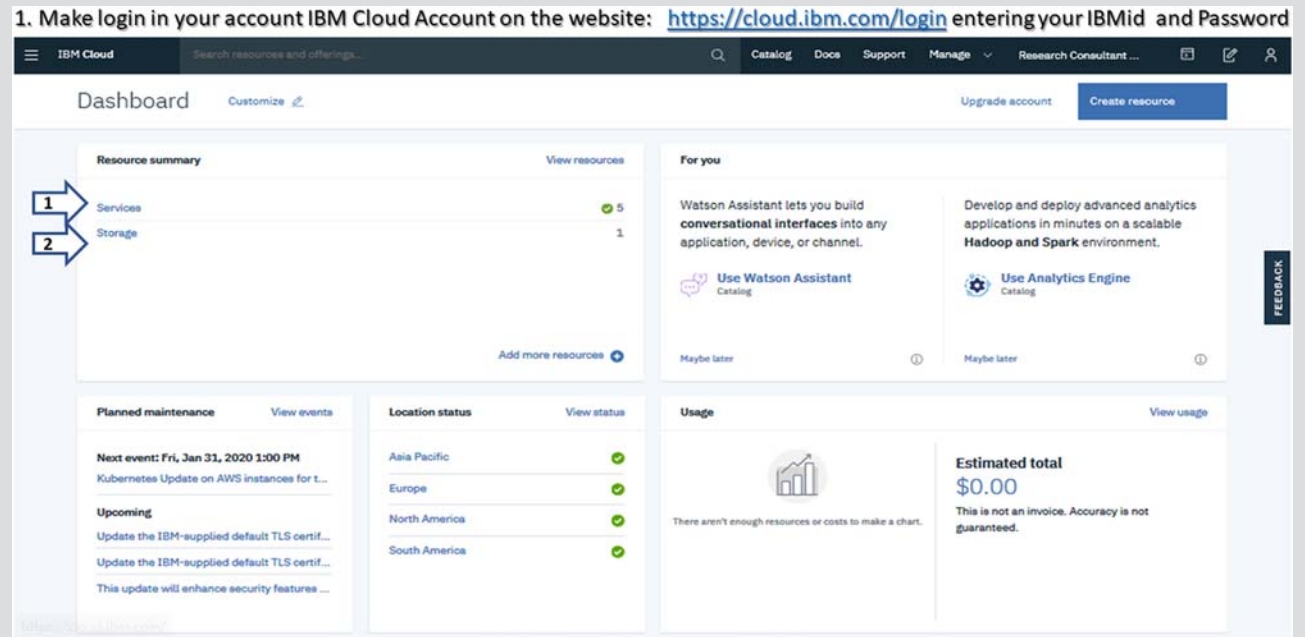
The steps to “Obtain ML models from IBM Watson Auto AI experimenter for a Breast Cancer dataset and deploy the best model” are summarized in Table of slides 4.4, and each step of the example is visually explained using screen sequences with instructions in figures.

Note: The “IBM Cloud website” is continuously evolving every day by consequence the IBM CLOUD website screens change frequently, some screens could be updated to optimize and simplify the procedures. I recommend understanding very well the objectives, applying them accordingly with the new screen’s formats shown at this table and the current IBM Cloud website contents. This research uses the “IBM Cloud Pak for Data” which is a fully integrated data and AI Watson platform.

Table of slides 4.4 steps to “Obtain ML models from IBM SPSS Modeler Flow Auto AI algorithms for a Breast Cancer dataset and deploy the best model.”

Slide Description Screen figure

1 Login to your IBM Cloud Account at the website: <https://cloud.ibm.com/login> entering your assigned IBMid and Password
 Note: You must have the 5 services available and 1 storage created as per Chapter 3 and Chapter 4, then click the “Services”



You must have the 5 services available and 1 storage created on Chapter 3 and research on Chapter 4, then click the “Services”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 2 Description In the IBM Cloud—Resource list—verifying services Click to expand “Services(5)” and “Storage(1),” check the availability of the “Machine Learning service” necessary for this research and click the service “Watson Studio” to go to its service screen

Screen figure

2. In the IBM Cloud – Resource list – verifying services

Name	Group	Location	Offering	Status	Tags
Filter by name or IP address... Filter by group or org... Filter... Filter... Filter... Filter...					
Devices (0)					
VPC infrastructure (0)					
Clusters (0)					
Cloud Foundry apps (0)					
Cloud Foundry services (0)					
Services (5)					
Natural Language Understanding-ch2	Default	Dallas	Natural Language Understa...	Active	nlp
Speech to Text-ch2	Default	Dallas	Speech to Text	Active	nlp
Text to Speech-ch2	Default	Dallas	Text to Speech	Active	nlp
Watson Studio-co	Default	Dallas	Watson Studio	Active	—
pm-20-at	Default	Dallas	Machine Learning	Active	—
Storage (1)					
cloud-object-storage-yp	Default	Global	Cloud Object Storage	Provisioned	—
Network (0)					
Cloud Foundry enterprise environments (0)					

Click to expand “Services(5)” and “Storage(1)”, check the availability of the Machine Learning service necessary for this tutorial and click the service “Watson Studio” to go to the service screen

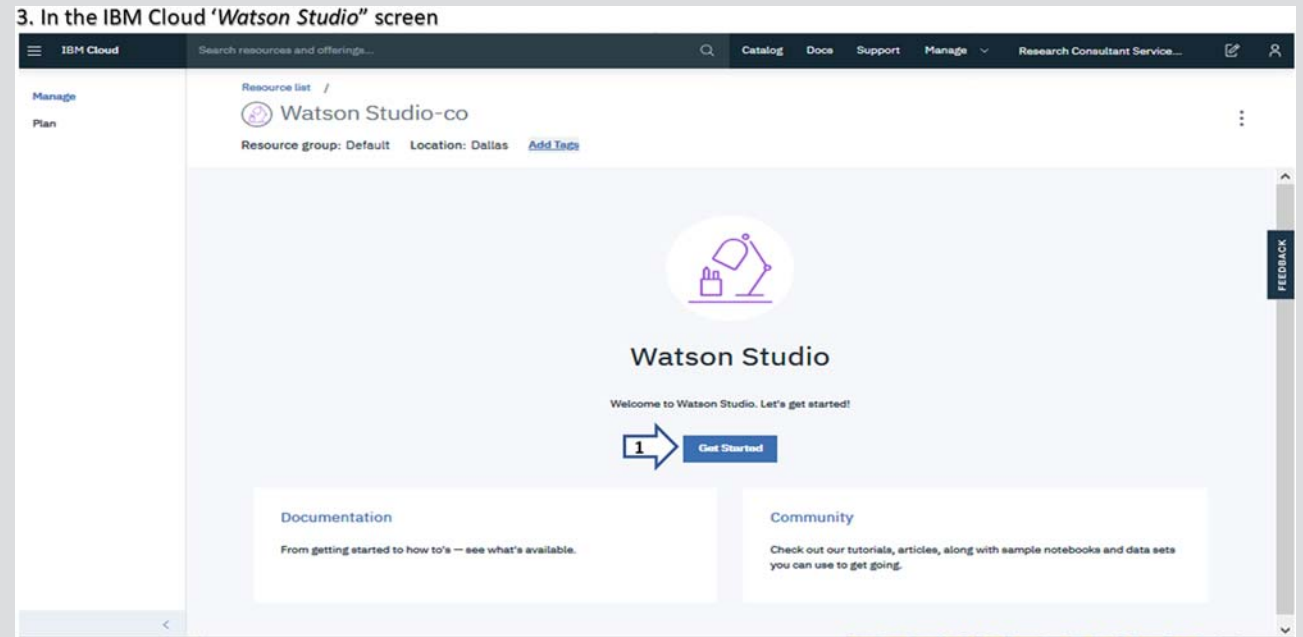
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description
3 In the IBM Cloud "Watson Studio"
screen
Press the button "Get Started"

Screen figure



Press the button "Get Started"

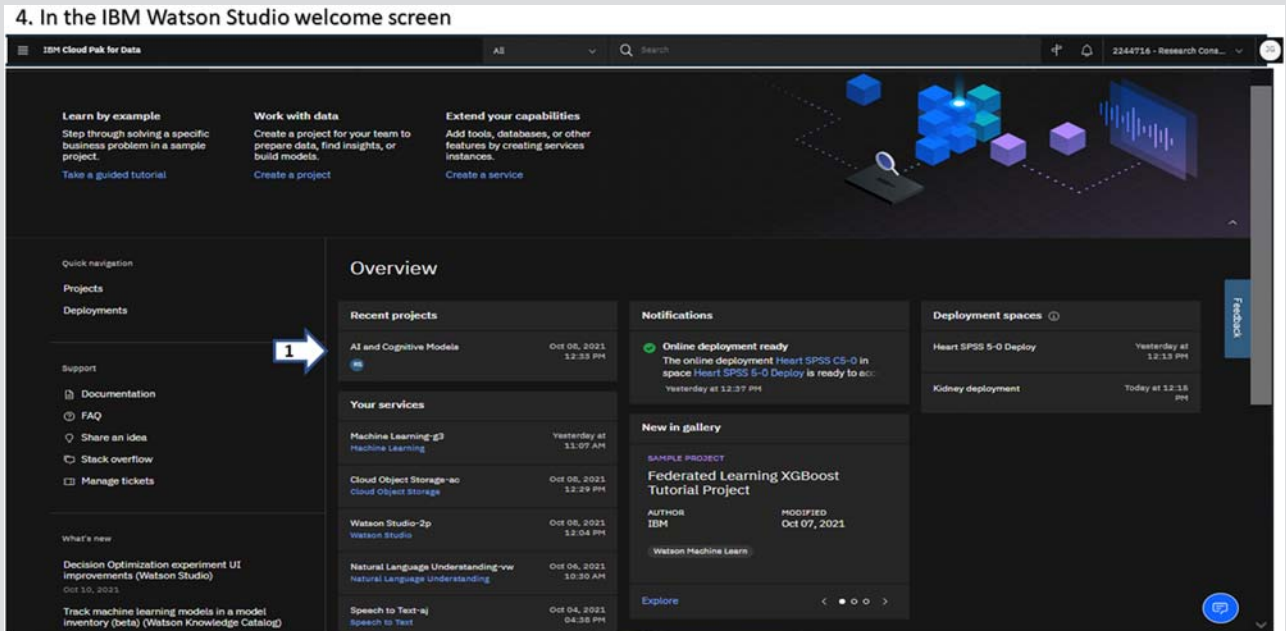
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 4 Description
In the IBM Watson Studio welcome screen
Click the recently updated project
"AI and Cognitive Models"

Screen figure



Click the recently updated project "AI and Cognitive Models"

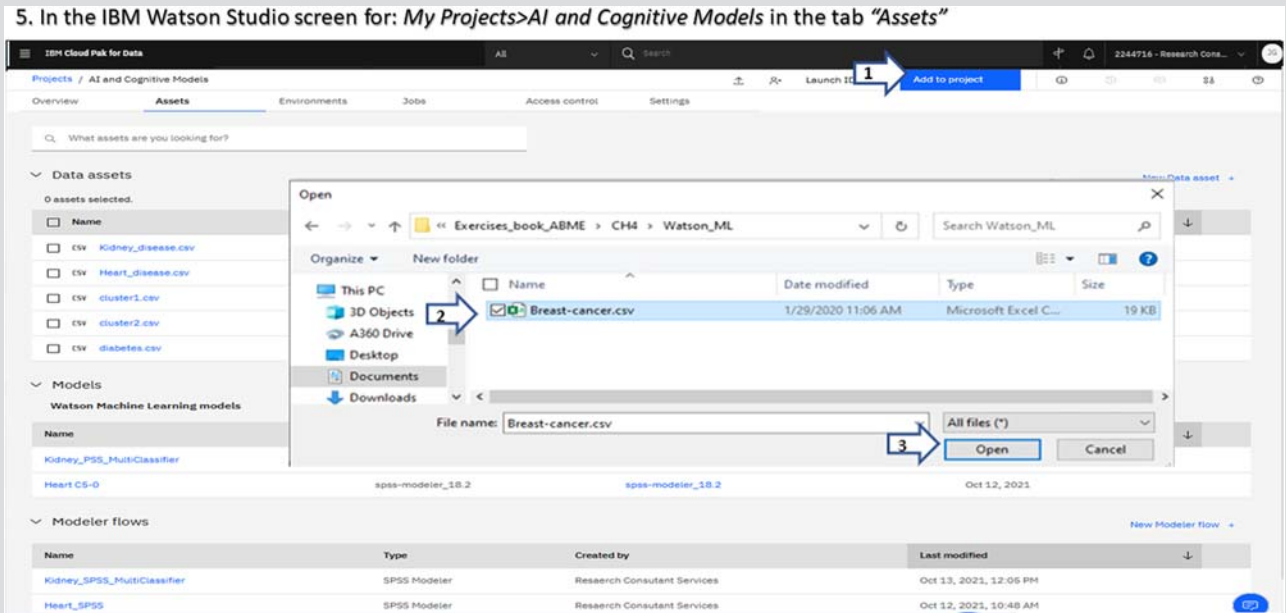
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 5 Description In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models* in the tab "Assets" Click on New Dataset and load asset using "browse," select the dataset "Breast-cancer.csv" in the data companion directory and press the button "Open"

Screen figure



Click on "New Dataset" and load asset using "browse", select the data set "Breast-cancer.csv" in the data companion directory and press the button "Open"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

6 In the IBM Watson Studio "My Projects / AI and Cognitive Models" section "Assets": observe that "Breast-cancer.csv" is available. Observe in "Data assets" section the "Breast_cancer.csv" is loaded and click on the button "Add to project"

6. IBM Watson Studio "My Projects / AI and Cognitive Models" section Assets: observe that asset "Breast_cancer.csv" is available

The screenshot shows the IBM Watson Studio interface. At the top, there is a navigation bar with "My Projects / AI and Cognitive Models" and a search bar. Below this, there are tabs for "Overview", "Assets", "Environments", "Jobs", "Deployments", "Access Control", and "Settings". The "Assets" tab is active, and the "Data assets" section is expanded. A table lists several data assets:

<input type="checkbox"/>	NAME	TYPE	CREATED BY	LAST MODIFIED	ACTIONS
<input checked="" type="checkbox"/>	CSV Breast-cancer.csv	Data Asset	Jorge Garza-Ulloa	29 Jan 2020, 11:09:37 am	
<input type="checkbox"/>	CSV Kidney_disease.csv	Data Asset	Jorge Garza-Ulloa	25 Jan 2020, 12:18:22 pm	
<input type="checkbox"/>	CSV Heart_disease.csv	Data Asset	Jorge Garza-Ulloa	21 Jan 2020, 4:42:36 pm	
<input type="checkbox"/>	cluster2	Data Asset	Jorge Garza-Ulloa	18 Jan 2020, 4:40:19 pm	
<input type="checkbox"/>	cluster1	Data Asset	Jorge Garza-Ulloa	18 Jan 2020, 4:39:36 pm	
<input type="checkbox"/>	CSV diabetes.csv	Data Asset	Jorge Garza-Ulloa	1 Jan 2020, 11:28:07 am	

Below the table, there are sections for "AutoAI experiments" and "Deep learning experiments", each with a "New" button. At the bottom, there is a "Models" section. On the right side, there is a "Drop files here or browse for files to upload" area with a "Stay on the page until upload completes. Incomplete uploads are cancelled" message.

Observe in "Data assets" section the "Breasy_cancer.csv" is loaded and click on the button "Add to project"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

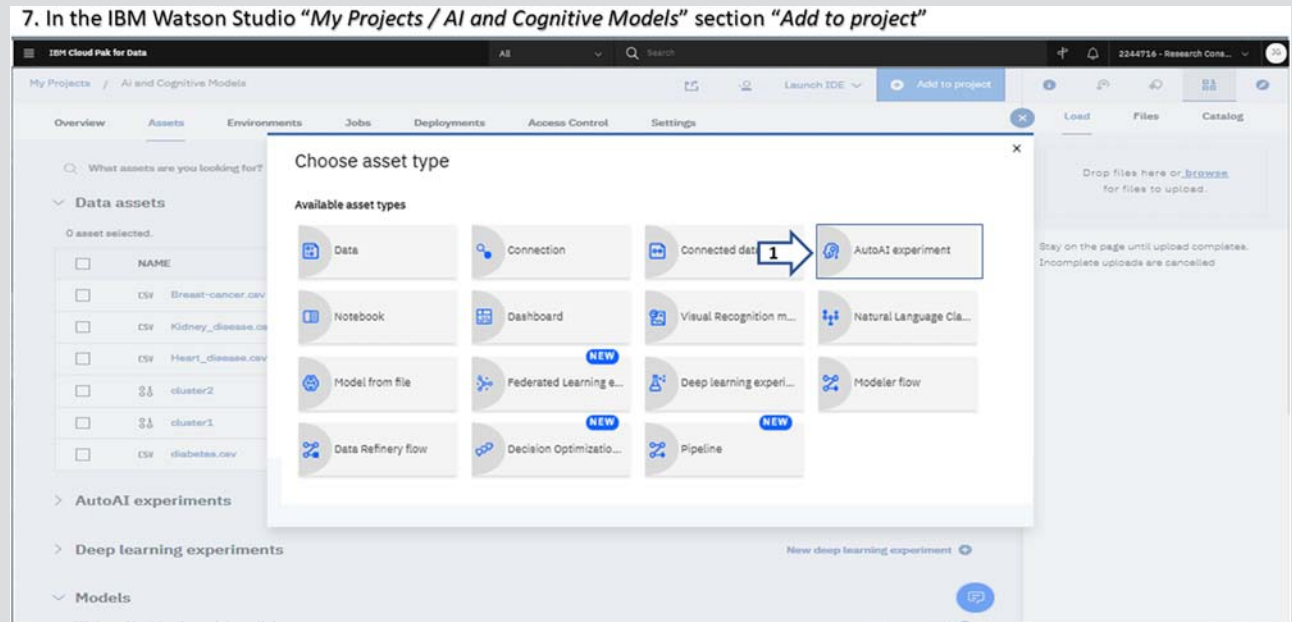
(Continued)

(Continued)

Slide Description

Screen figure

7 In the IBM Watson Studio “My Projects/AI and Cognitive Models” section “Add to project” Click on the button “Auto AI experiment”



Click on the button “Auto AI experiment”

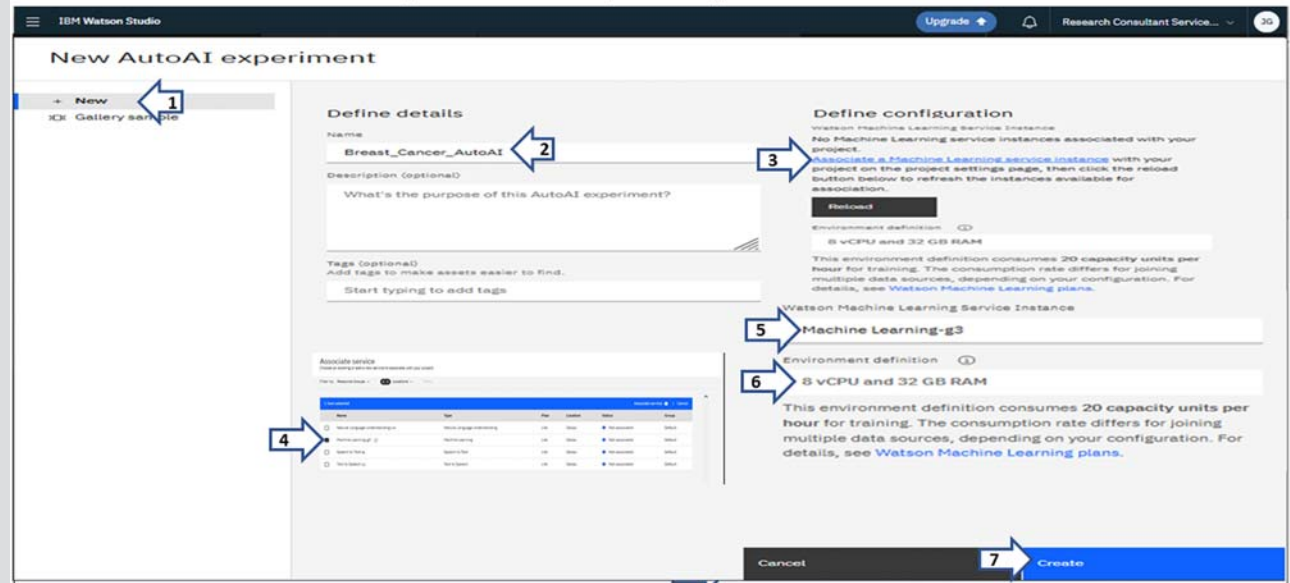
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 8 Description In the IBM Watson Studio—AutoAI Experiment screen: “Experiment type” Select “New”, Name = “Breast_Cancer_AutoAI”, in Define configuration click on “Associate a Machine Learning instance” and select your “Assign Machine Learning”. For Environment definition = “8 vCPU and 32 GB RAM”, and click on the button “Create”

Screen figure 8. In the IBM Watson Studio – AutoAI Experiment screen: “Experiment type”



Select “New”, Name = “Breast_Cancer_AutoAI”, in Define configuration click on “Associate a Machine Learning instance” and Select your “Assign Machine Learning”. For Environment definition = “8 vCPU and 32 GB RAM”, and click on the button “Create”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide **Description**

9 In the IBM Watson Studio “My Projects/AI and Cognitive Models/Breast-cancer_AutoAI screen”: Loading data
Click on “Browse”, from windows dialog select “Breast-cancer.csv” at the sub-directory indicate and click on the button “Open”

Screen figure

9. In the IBM Watson Studio My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen”: Loading data

Click on “Browse”, from windows dialog select “Breast-cancer.csv” at the sub-directory indicate and click on the button “Open”

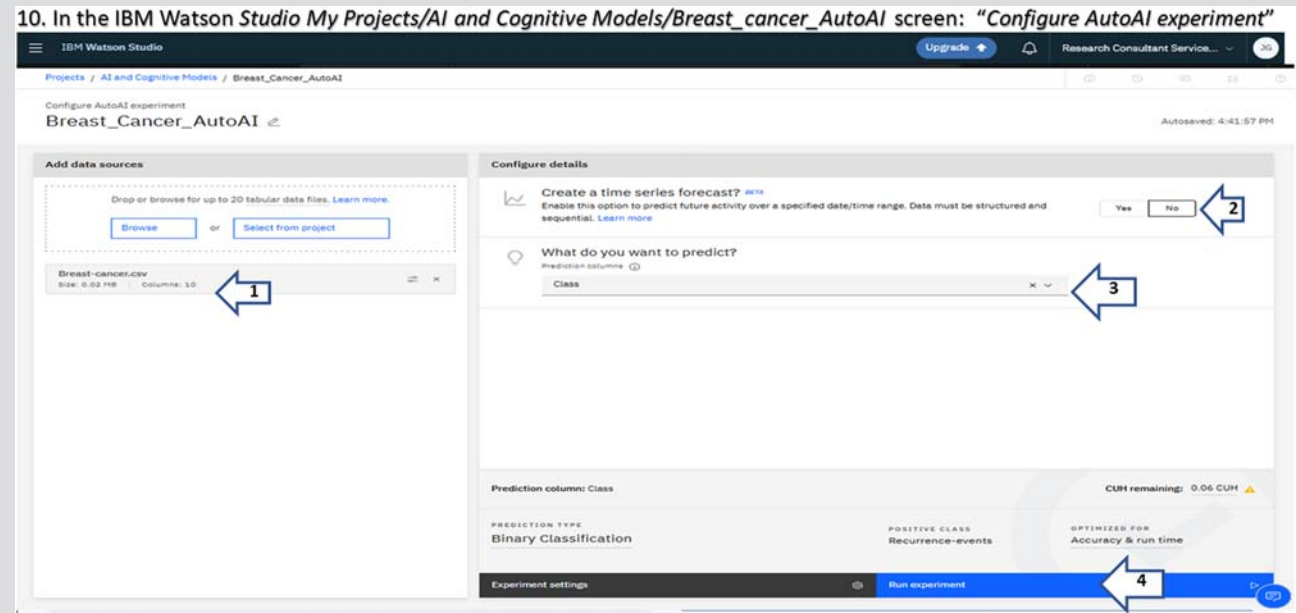
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 10 Description In the IBM Watson Studio “My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen”: “Configure AutoAI experiment” Verify the “Breast-cancer.csv” is loaded, on *Configure details* select “No” for “Create a time series forecast”, on *What to you want to predict* select the field “Class” and click the button “Run Experiment”

Screen figure



Verify the “Breast-cancer.csv” is loaded, on *Configure details* select “No” for “Create a time series forecast”, on *What to you want to predict* select the field “Class”, and click the button “Run Experiment”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

11 In the IBM Watson Studio "My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen": "Run AutoAI Experiment"
Note: Using the free version "Lite" will appear a windows "Experiment exceeds compute Capacity", press "here" to visualize the Capacity Unit Hour that indicate "19.9 CUH" remaining only "0.1" for this month. We need to "wait for next month or upgrade to a non-free version".

11. In IBM Watson Studio My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen: "Run AutoAI Experiment"

The screenshot displays the IBM Watson Studio interface for a project named "Breast_cancer_AutoAI". A modal dialog box is open in the center, titled "Experiment exceeds compute capacity". The message states: "This experiment exceeds the compute capacity allotted by the service plan. Pipelines created to this point can be saved, but no more pipelines will be generated until the service owner upgrades the Watson Machine Learning service plan for instance Machine Learning-g3. Then you can run the experiment. If you are not the owner, contact the owner to request an upgrade. Learn more". Below the message is a "Cancel" button. In the background, a "Pipeline leaderboard" table is visible, showing usage for "Machine Learning-g3".

Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Enhancements	Build time
Pipeline generation was stopped					
Watson Machine Learning CUH usage this month					
	Machine Learning-g3		19.9 CUH used this month		0.1 CUH remaining

Note: Using the free version "Lite" will appear a windows "Experiment exceeds compute Capacity", press "here" to visualize the Capacity Unit Hour that indicate "19.9 CUH" remaining only "0.1" for this month. We need to "wait for next month or upgrade to a non-free version"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

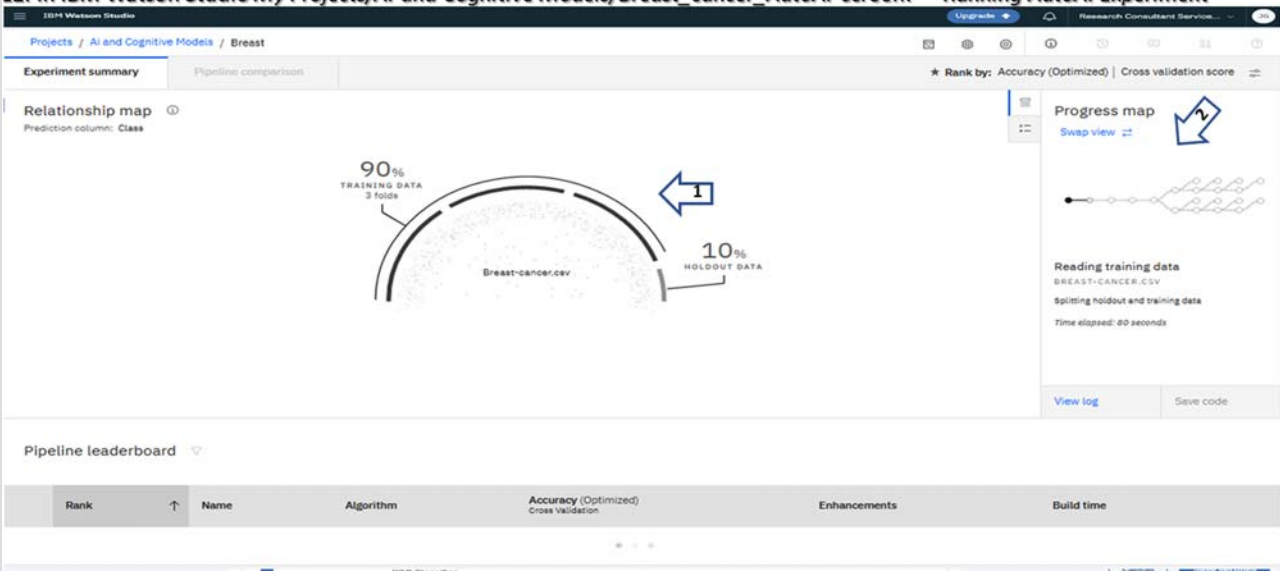
(Continued)

(Continued)

Slide 12 Description IBM Watson Studio "My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen": "Running AutoAI Experiment" The non-free version for "Run AutoAI Experiment" test many AI models in the following minutes, the "data partition 90% as training data" and "10% as Holdout data or test data", on the top right side shows a "Progress Map"

Screen figure

12. In IBM Watson Studio My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen: "Running AutoAI Experiment"



The screenshot displays the IBM Watson Studio interface for an AutoAI experiment. At the top, the breadcrumb navigation shows 'Projects / AI and Cognitive Models / Breast'. Below this, there are tabs for 'Experiment summary' and 'Pipeline comparison'. The main area features a 'Relationship map' with a 'Prediction column: Class'. A circular diagram illustrates data partitioning: a large arc represents '90% TRAINING DATA 3 folds' and a smaller arc represents '10% HOLDOUT DATA'. A blue arrow labeled '1' points to the holdout data. Below the diagram is a 'Pipeline leaderboard' table with columns: Rank, Name, Algorithm, Accuracy (Optimized) Cross Validation, Enhancements, and Build time. On the right side, a 'Progress map' shows a workflow diagram and a 'Reading training data' section with details: 'BREAST-CANCER.CSV', 'Splitting holdout and training data', and 'Time elapsed: 00 seconds'. A 'View log' button is also present.

The non-free version for "Run AutoAI Experiment" test many AI models in the following minutes, the "data partition 90% as training data" and "10% as Holdout data or test data", on the top right side shows a "Progress Map"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 13 Description
In the IBM Watson Studio screen: "My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen": "Running AutoAI Experiment"
Observe the "Experiment details" that begin to training and testing different "AI algorithm", this is indicate at the top left as a "Relation ship map", at the top right as "Progress map", and at the bottom at the "Pipeline leaderboard" indicate the "AI algorithm Accuracy" in descending order.

Screen figure

13. In IBM Watson Studio My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen: "Running AutoAI Experiment"

The screenshot displays the IBM Watson Studio interface for an AutoAI experiment. At the top, there is a navigation bar with 'IBM Watson Studio' and 'Upgrade' options. Below this, the 'Experiment summary' and 'Pipeline comparison' tabs are visible. The main area is divided into three sections: a 'Relationship map' on the left, a 'Progress map' on the right, and a 'Pipeline leaderboard' at the bottom. The 'Relationship map' shows a semi-circular diagram with 'FEATURE TRANSFORMERS' at the top, 'PIPELINES' in the middle, and 'TOP ALGORITHMS' at the bottom. A blue arrow labeled '1' points to the 'PIPELINES' section. The 'Progress map' shows a sequence of steps with a blue arrow labeled '2' pointing to the 'Hyperparameter optimization' step. The 'Pipeline leaderboard' is a table with columns for Rank, Name, Algorithm, Accuracy (Optimized) Cross Validation, Enhancements, and Build time. A blue arrow labeled '3' points to the 'Accuracy (Optimized) Cross Validation' column. The table lists three pipelines, with Pipeline 3 having the highest accuracy of 0.730.

Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Enhancements	Build time
★ 1	Pipeline 3	Extra Trees Classifier	0.730	HPO-L, FE	00:00:18
2	Pipeline 1	Extra Trees Classifier	0.705	None	00:00:01
3	Pipeline 2	Extra Trees Classifier	0.705	HPO-L	00:00:09

Observe the "Experiment details" that begin to training and testing different "AI algorithm", this is indicate at the top left as a "Relation ship map", at the top right as "Progress map" , and at the bottom at the "Pipeline leaderboard" indicate the "AI algorithm Accuracy" in descending order..

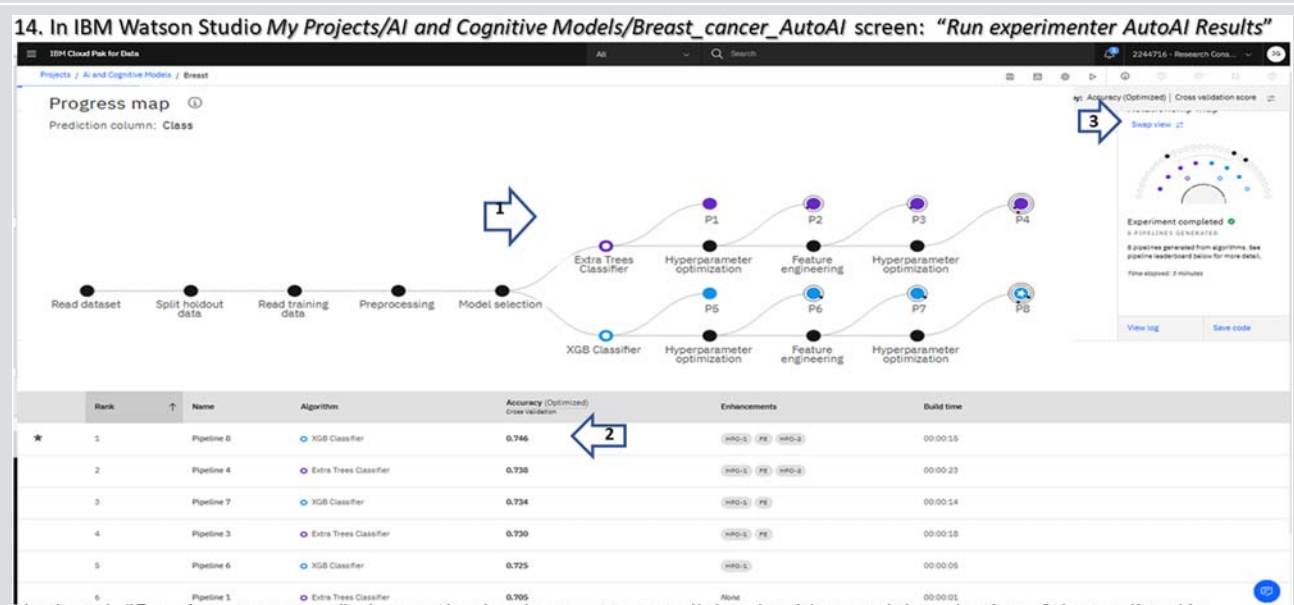
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 14 Description In the IBM Watson Studio screen: "My Projects/AI and Cognitive Models Breast_cancer_AutoAI screen": "Run experimenter AutoAI Results" In the tab "Experiment summary" observe the development to test all the algorithms and the selection of the top four listed in "Pipeline leaderboard," where the top algorithm is the "XGB Classifier" with an "Accuracy (optimized) of 0.746" Press click in "Swap view" of "Progress map" of "Progress map"

Screen figure 14. In IBM Watson Studio My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen: "Run experimenter AutoAI Results"



In the tab "Experiment summary" observe the development to test all the algorithms and the selection of the top listed in "Pipeline leaderboard", where the top algorithm is the "XGB Classifier" with a "Accuracy (optimized) of 0.746". Press click in "Swap view" of "Progress map"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 15 Description
In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Breast_cancer_AutoAI* screen: *“Saving Top AI Algorithm”*
In the *“Pipeline leaderboard”* select the one of first place *“XGB Classifier”*, and click on *“Save as”*

Screen figure

15. In IBM Watson Studio *My Projects/AI and Cognitive Models/Breast_cancer_AutoAI* screen: *“Saving Top AI Algorithm”*

Relationship map

Experiment completed
3 PIPELINES GENERATED
3 pipelines generated from algorithm. See pipeline leaderboard below for more detail.
Time elapsed: 2 minutes

View log Save code

Pipeline leaderboard

Rank	Name	Algorithm	Accuracy (Optimized) Cross Validation	Enhancements	Build time
★ 1	Pipeline 0	XGB Classifier	0.746	HPO-1 FE HPO-2	00:00:16
2	Pipeline 4	Extra Trees Classifier	0.738	HPO-1 FE HPO-2	00:00:23
3	Pipeline 7	XGB Classifier	0.734	HPO-1 FE	00:00:14

Save as

In the *“Pipeline leaderboard”* select the one of first place *“XGB Classifier”* , and click on *“ Save as”*

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

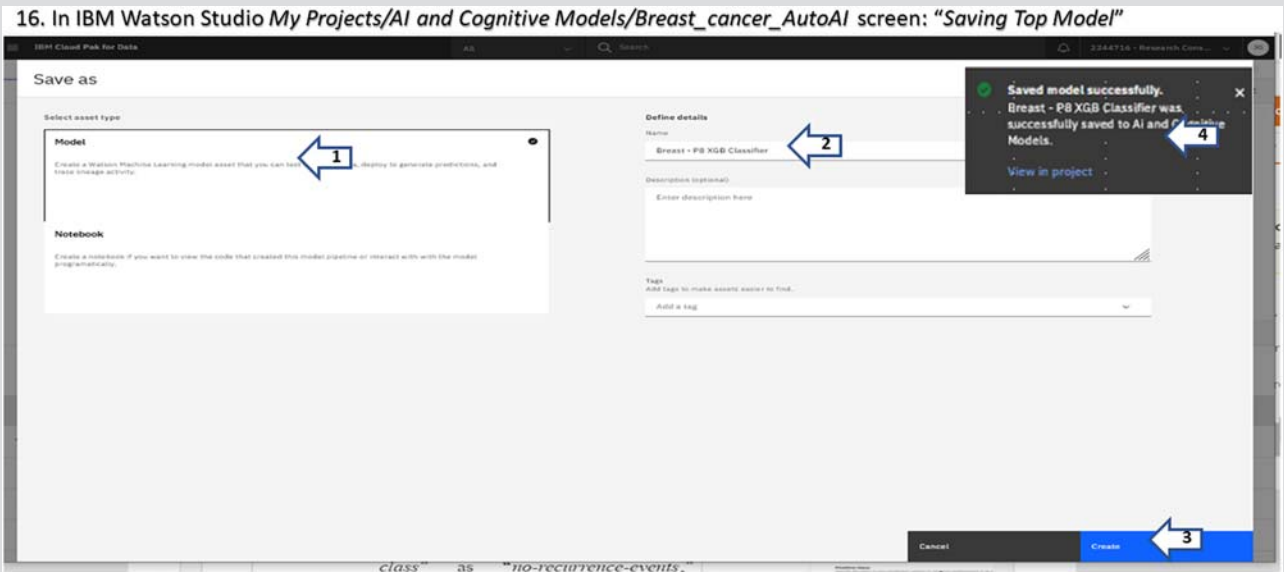
(Continued)

(Continued)

Slide Description

Screen figure

16 In the IBM Watson Studio screen: *My Projects/AI and Cognitive Models/ Breast_cancer_AutoAI* screen: : “Saving Top Model”
At the window “Save as model” select “Model”, then define model name as “Breast-P8 XGB Classifier”, click on “Create”. Then a small windows indicated that “Saved model successfully” and finally click on “View in project”



At the window “Save as model” select “Model”, then define model name as “Breast-P8 XGB Classifier”, click on “Create”. Then a small windows indicated that “Saved model successfully” and finally click on “View in project”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 17
Description
In IBM Watson Studio My Projects/
AI and Cognitive Models/
Breast_cancer_AutoAI screen:
Promote to deployment space
In the model "Breast P8 XGB
Classifier," verify the "Schema"
and "Type" then click on "Promote
to deployment space" as shown

Screen figure

17. In IBM Watson Studio My Projects/AI and Cognitive Models/Breast_cancer_AutoAI screen: Promote to deployment space

Column	Type
age	"string"
breast	"string"
breast-quad	"string"
deg-malign	"integer"
inv-nodes	"string"
inv-size	"string"
metapath	"string"
node-caps	"string"
num-nodes	"string"

In the model "Breast_P8 XGB Classifier", verify the "Schema" and "Type" then click on "Promote to deployment space" as shown

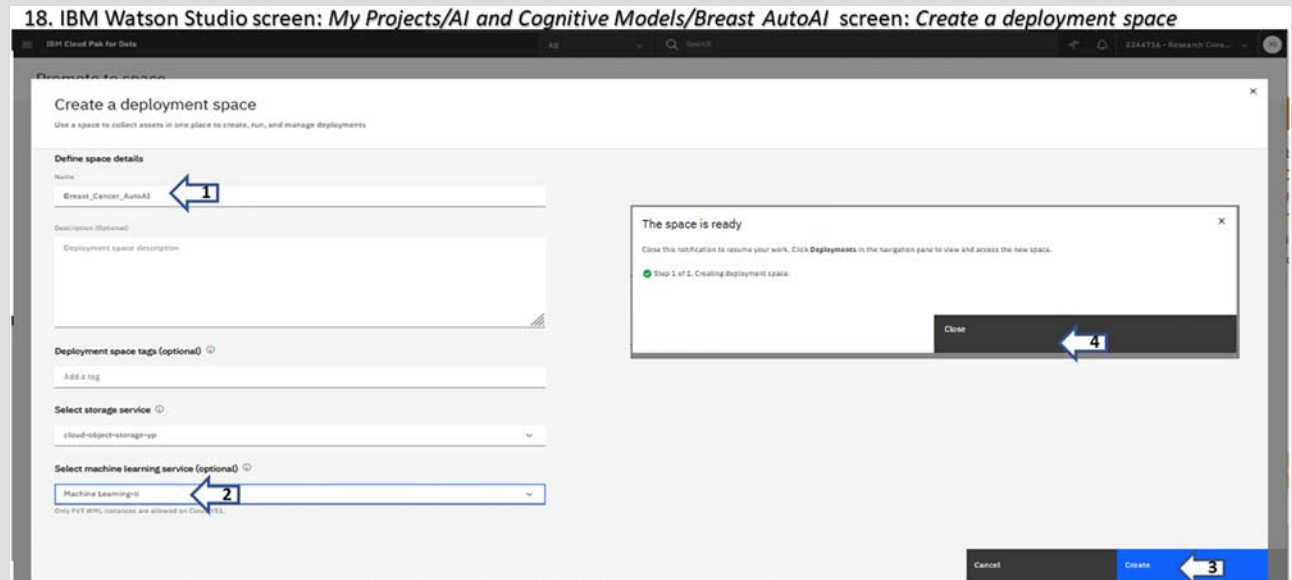
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 18 Description
IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Breast_cancer_AutoAI* - Create a deployment space
In the “Create a deployment space” specify name as “Breast_Cancer_AutoAI” and select storage service “your assigned cloud object storage”, select machine learning services as “your assigned machine learning service”, and finally click at “Create” button. A confirmation windows will indicate “The space is ready” and finally click “Close”

Screen figure



In the “Create a deployment space” specify name as “Breast_Cancer_AutoAI” and select storage service “your assigned cloud object storage”, select machine learning services as “your assigned machine learning service”, and finally click at “create” button. A confirmation windows will indicate “The space is ready” and finally click “Close”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

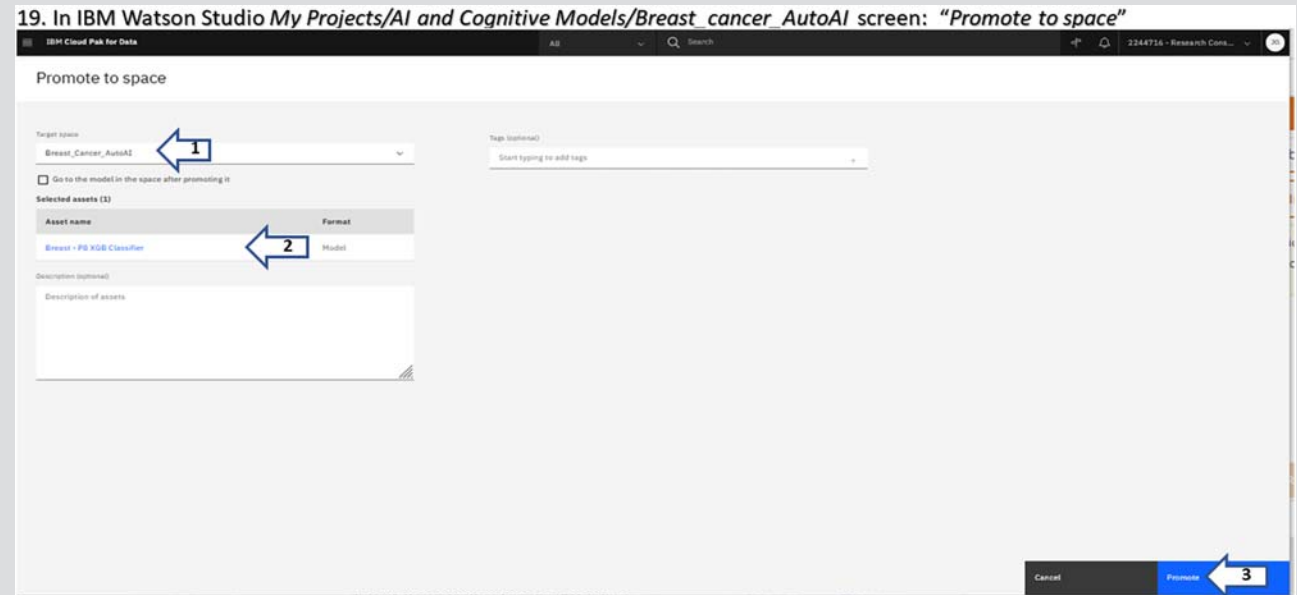
(Continued)

Slide **Description**

19 In IBM Watson Studio *My Projects/ AI and Cognitive Models/ Breast_cancer_AutoAI* screen: *"Promote to space"*

In *"Promote to space"* windows verify to select the target space as *"Breast_cancer_AutoAI"* and asset name as *"Breast-P8 XGB Classifier"*. Then click on *"Promote"* as indicated in the slide

Screen figure



In *"Promote to space"* windows verify o select the target space as *"Breast_cancer_AutoAI"* and asset name as *"Breast-P8 XGB Classifier"*. Then click on *"Promote"* as indicated in the slide.

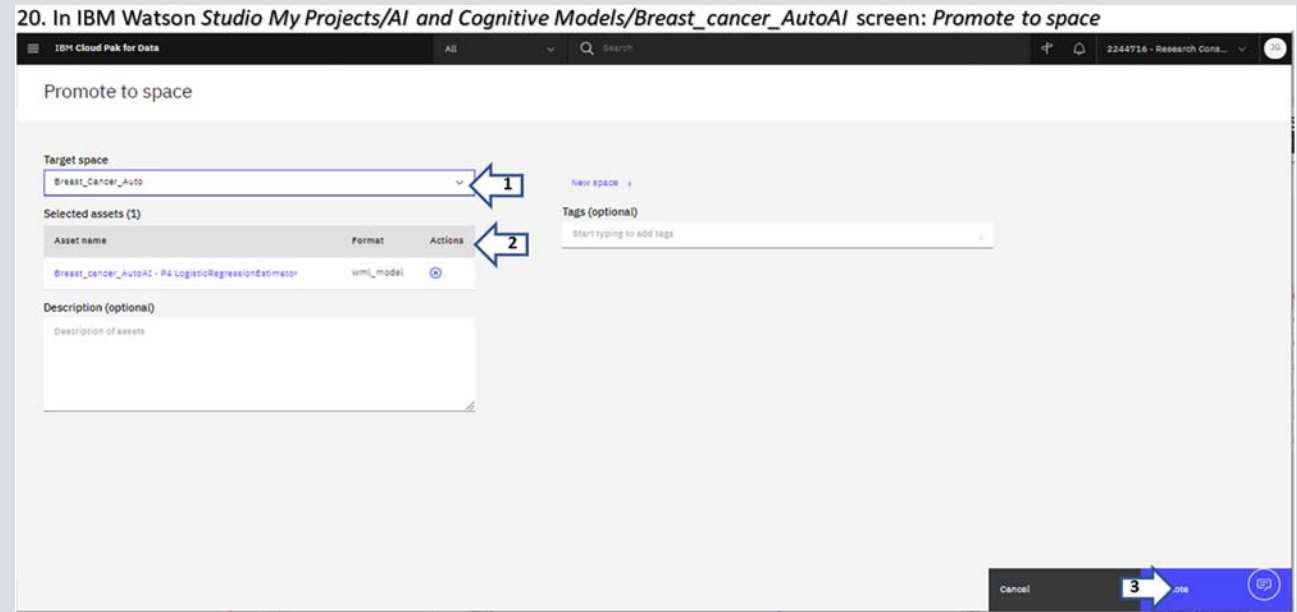
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 20 Description In IBM Watson Studio My Projects/ AI and Cognitive Models/ Breast_cancer_AutoAI screen Promote to space In the “Promote to space” select “Target space as Breast_Cancer_Auto,” “Select assets as Breast_cancer_Auto” and click on the “Promote” button as indicated

Screen figure



In the “Promote to space” select “Target space as Breast_Cancer_Auto”, “Select assets as Breast_cancer_AutoAI-P4 LogisticRegressionEstimator” and click on the “Promote” button as indicated.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

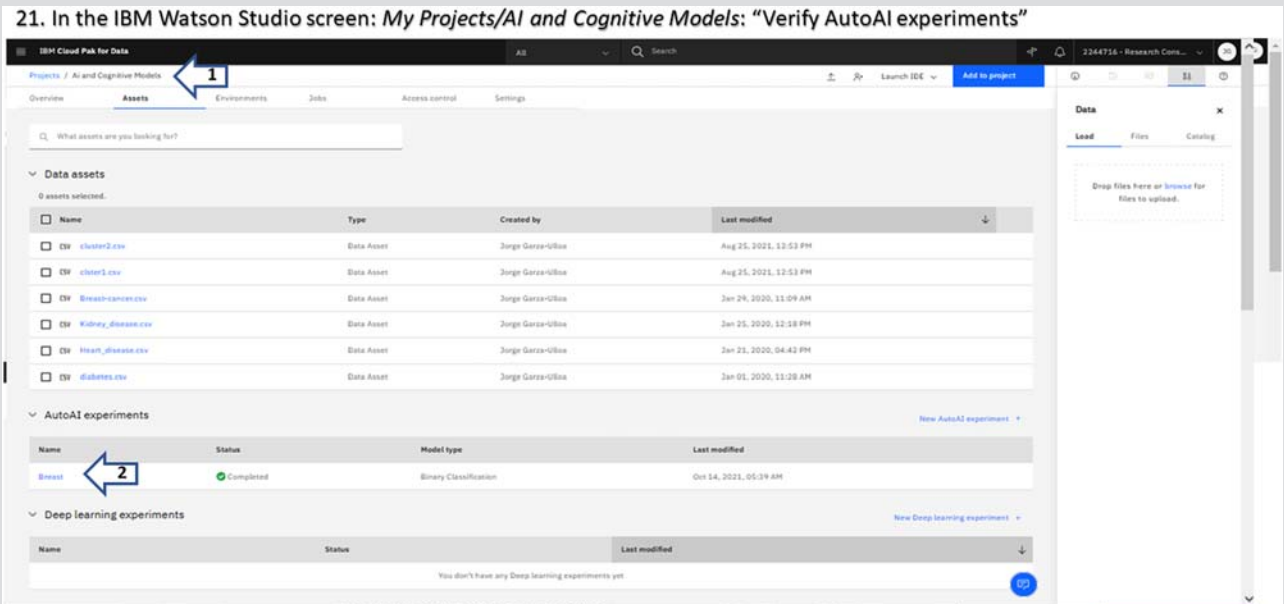
(Continued)

(Continued)

Slide Description

Screen figure

21 In the IBM Watson Studio screen: My Projects/AI and Cognitive Models: "Verify AutoAI experiments"
Select Project/AI and Cognitive Models, and verify at "Auto AI experiments" has stored "Breast" experiment



Select Project/AI and Cognitive Models, and verify at "Auto AI experiments" has stored "Breast" experiment

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

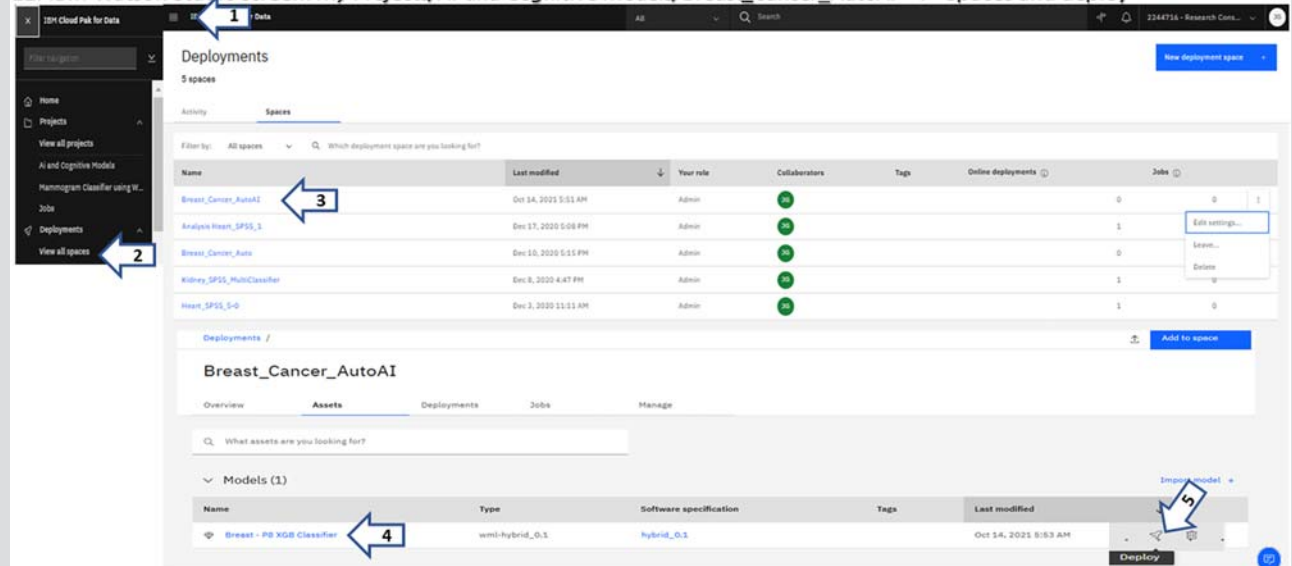
(Continued)

(Continued)

Slide Description Screen figure

22 IBM Watson Studio screen: My Projects/AI and Cognitive Models/Breast_cancer_AutoAI: "Spaces and deploy"
Click in the "Navigation menu" as shown in the slide and select "Deployment > View all spaces," then at the tab "Deployment" click space for "Breast_cancer_AutoAI" as shown in the upper screen. In the Model "Breast_cancer_AutoAI- P8 XGB Classifier" in the action select "Deploy icon" as indicated in the lower screen

22. IBM Watson Studio screen: My Projects/AI and Cognitive Models/Breast_cancer_AutoAI - : "Spaces and deploy"



Click in the "Navigation menu" as shown in the slide and select "Deployment > View all spaces", then at the tab "Deployment" click on the space for "Breast_cancer_AutoAI" as shown in the upper screen. Select the Model "Breast - P8 XGB Classifier" and click on "Deploy icon" as indicated in the lower screen.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

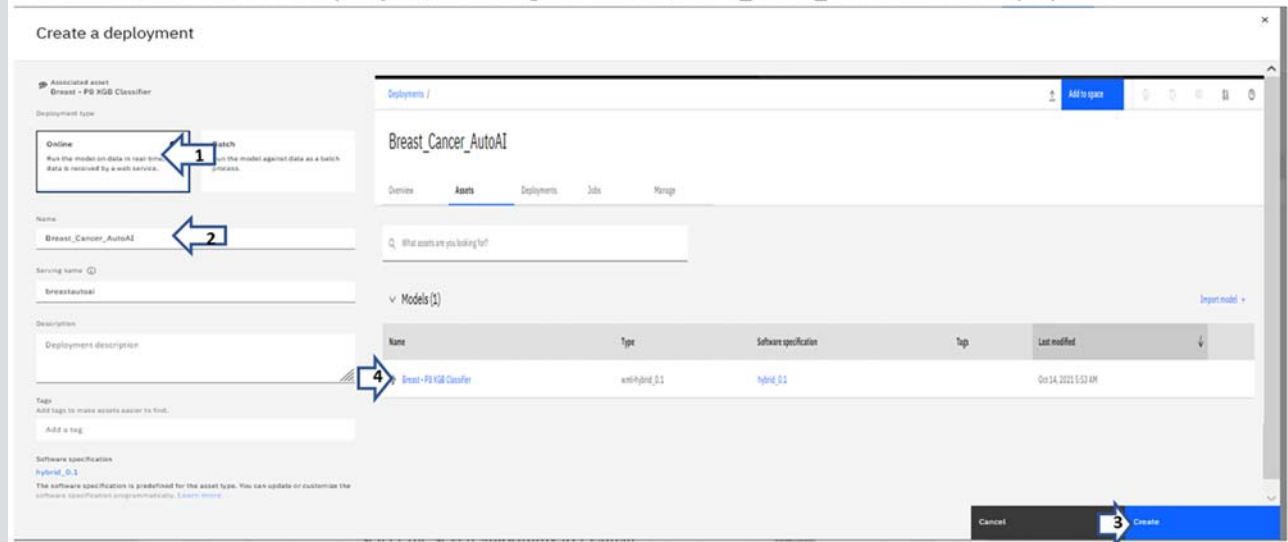
(Continued)

(Continued)

Slide 23 Description
IBM Watson Studio screen: My Projects/AI and Cognitive Models/Breast_cancer_AutoAI: "create deployment"
Select the option: "Deployments" "Online," then click in the deployment name "Breast_cancer_AutoAI" and click on "Create" button as shown in the upper screen. Click in the model "Breast_cancer-P8 XGB Classifier" as indicated in the top right screen

Screen figure

23. IBM Watson Studio screen: My Projects/AI and Cognitive Models/Breast_cancer_AutoAI: "create deployment"



Select the option: "Deployments" "Online", then click in the deployment name "Breast_cancer_AutoAI" and click on "Create" button as shown in the upper screen. Click in the model "Breast_-P8 XGB Classifier" as indicated in the top right screen.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

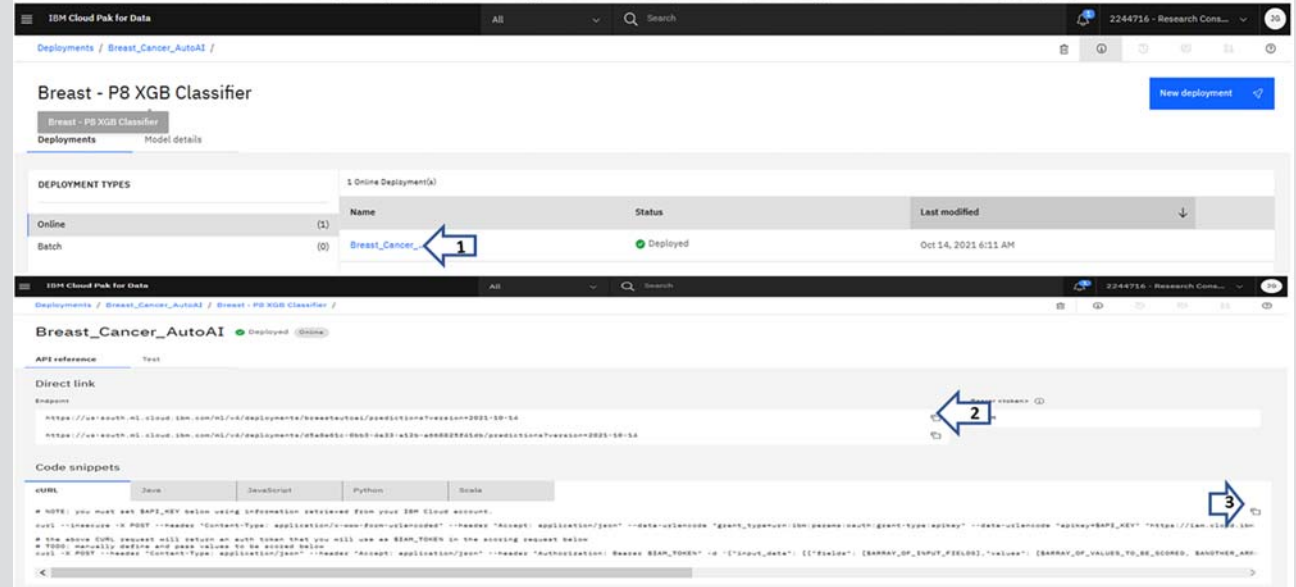
(Continued)

(Continued)

Slide 24 Description
IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Breast_cancer_AutoAI - ...: "Using deployment"*
Select the option: "Deployments" "Online," then click in the deployment name "Breast_cancer_..." as shown in the top screen. You can copy the "Bearer <Token>" and make the deployment with: "cURL," "Java," "JavaScript," or "Python." Select "Test" as indicated in the lower screen

Screen figure

24. IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Breast_cancer_AutoAI - ...: "Using deployment"*



Select the option: "Deployments" "Online", then click in the deployment name "Breast_cancer_..." as shown in the top screen. You can copy the "Bearer <Token>" and make the deployment with: "cURL," "Java", "JavaScript" or "Pyhton". Select Test as indicated In the lower screen. From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

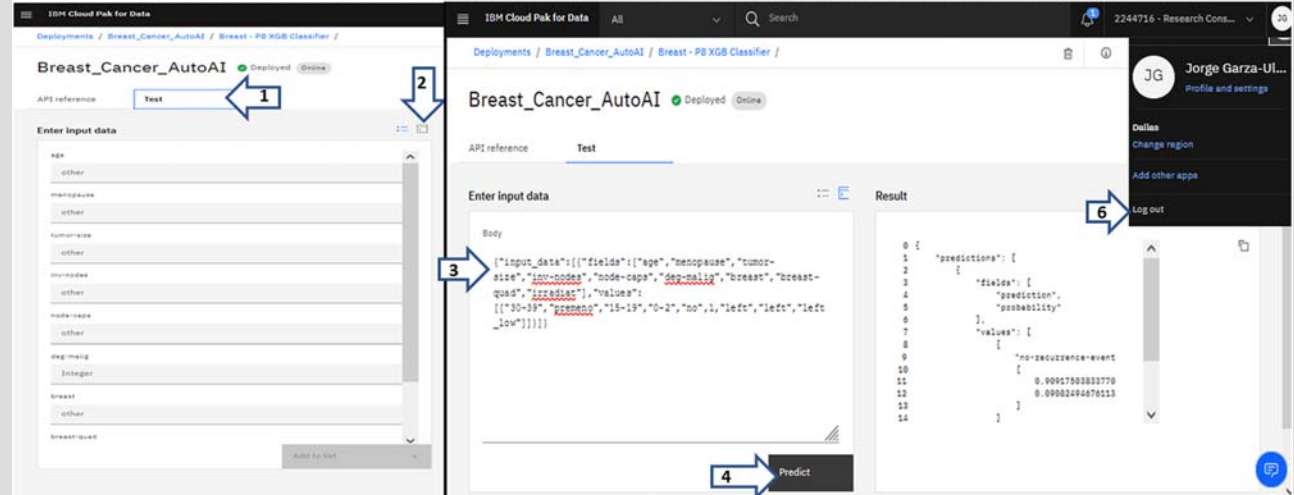
(Continued)

Slide 25 Description Screen figure

IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Breast_cancer_AutoAI "Testing deployment"*

In option "Test" select the JSON input data and copy the following instruction: `{"input_data":{"fields":["age","menopause","tumor-size","inv-nodes","node-caps","deg-malig","breast","breast-quad","irradiat"],"values":["30-39","premeno","15-19","0-2","no",1,"left","left","left_low"]}}`. Then click on the button "Predict" and the predict for "class = no-recurrence-events" with "probability = 0.909" and "confidence value = 0.01". Click at the top right and "logout"

25. IBM Watson Studio screen: *My Projects/AI and Cognitive Models/Breast_cancer_AutoAI* screen: "Testing deployment"



In option "Test" select the JSON input data and copy the following instruction: `{"input_data":{"fields":["age","menopause","tumor-size","inv-nodes","node-caps","deg-malig","breast","breast-quad","irradiat"],"values":["30-39","premeno","15-19","0-2","no",1,"left","left","left_low"]}}`. Then click on the button "Predict" and the predict for "class=no-recurrence-events" with "probability= 0.909" and "confidence value = 0.09". Click at the top right and "logout"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

We can have the following conclusions on “Obtain ML models from IBM Watson Auto AI experimenter for a Breast Cancer dataset and test the best model.”

- It was possible to predict when patient has recurrent “breast cancer” based on the attributes “class” of the given dataset; the top performance model was the “XBC Classifier” selected from “Pipeline” with a “Accuracy (optimized) of 0.746”
- The deployment to test the model is simplified and tested in a “JSON” form.
 - The “AutoAI” allows one to get started quickly with “experimentation,” “evaluation,” and “deployment.”
 - The “pipeline leaderboard” allows a fast model selection from the top-performing ML models.

Recommendations

- A larger dataset will help to analyze with more precision the “class” with more variations on the predicted value.
- Incorporate in the dataset, or create a new dataset that includes, more new attributes that will help in the prediction, such as “Body Mass Index,” “Breast density,” “Lack of exercise,” “Alcoholic drink,” “Previous medical treatment,” “Gender,” and others factors. These new attributes have the objective of helping determine the type and subtype of “breast cancer.”
- Try to include the type of recurrent breast cancer as: “local recurrence,” “regional recurrence,” and “distant recurrence” [28–30].

Note: For the next research tutorial we will apply MATLAB ML solution: Statistics and Machine Learning Toolbox.

The “Statistics and Machine Learning Toolbox” [31] provides functions and apps to describe, analyze, and model data. From “descriptive statistics” and “plots for exploratory data analysis,” “fit probability distributions to data,” “perform hypothesis tests.” “ML” allows the application of “regression and classification algorithms” using its own applications to draw inferences from data and to build “predictive models.” The toolbox provides “supervised and unsupervised machine learning” algorithms, including “support vector machines,” “boosted and bagged decision trees,” “k-nearest neighbor,” “k-medoids,” “hierarchical clustering,” “Gaussian mixture models,” and “hidden Markov” models. This section is focused on “ML predictive models” using “MATLAB: Statistics and Machine Learning Toolbox.”

4.12.2.5 Research tutorial 4.5 MATLAB: Statistics and Machine Learning Toolbox for a “Diabetes dataset AI modeling for Classifier Model and a Regression Model”

4.12.2.5.1 Case for research

“Obtain AI Models using MATLAB: Statistics and Machine Learning Toolbox for a diabetes dataset.”

4.12.2.5.2 General objective for this research

Obtain two “AI Machine Learning predictive Models” for a diabetes dataset, one “Classifier model” and one “Regression model” using the following MATLAB applications: “Classification Learner” and “Regression Learner MATLAB” from the “Statistics and Machine Learning Toolbox.”

4.12.2.5.3 Specific objectives

- Obtain two “AI Machine Learning predictive Models” for a diabetes dataset, one as a “Classifier Model” and the other as a “Regression Model” using “Statistics and Machine Learning Toolbox” to “predict futures values for diabetes.”
- Obtain the best possible “Classifier model” using the “MATLAB Classification Learner” application from “Statistics and Machine Learning Toolbox.” Find “ML Classifier Models” with higher “accuracy,” verifying “Confusion Matrix,” “ROC Curve, and “Parallel coordinates plot.” “Export” the best model and test the prediction results using a special dataset for testing.
- Obtain the best possible “Regression model” using the “MATLAB Regression Learner” application from “Statistics and Machine Learning Toolbox.” Find “ML Regression Models” with lower “RMSE” Root Mean Square Error (RMSE) as the standard deviation of the residuals (prediction errors) (the lower is the best), obtaining a “Response Plot” of “pedi” (diabetes pedigree function) then “export” the best model and test the prediction results using a special dataset for testing.

4.12.2.5.4 Dataset

The dataset “diabetes.csv” is based on information in the Pima Indians Diabetes Database at the “National Institute of Diabetes and Digestive and Kidney Diseases.” It has “768 records,” with “8 fields (attributes)” as indicated in Table 4.4.

Note: This “diabetes” dataset was analyzed in Section 3.3.2 using IBM Watson SPSS Modeler Flow.

4.12.2.5.5 Procedure

The steps to obtain two “AI Machine Learning predictive Models” for a “diabetes dataset,” one as “Classifier Model” and the other as a “Regression Model” using the following MATLAB applications: “Classification Learner” and “Regression Learner” that are included in MATLAB: “Statistics and Machine Learning Toolbox,” are summarized in Table of slides 4.5 and each step of the example is visually explained using screen sequences with instructions in figures.

TABLE 4.4 Diabetes dataset to obtain ML Models using MATLAB: Statistics and Machine Learning Toolbox.

Field	Description* 768 instances of patients. All females >21 and <81 years old
preg	Number of times pregnant (integers)
plas	Plasma glucose concentration 2 h oral glucose tolerance test
pres	Diastolic blood pressure in mm Hg (real number)
skin	Triceps skin fold thickness in mm (real number)
Insu	2-h serum insulin in μ U/mL (real number)
mass	Body mass index based on weight in kg/(height in m) ² (real number)
pedi	Diabetes pedigree function (real number). It is a function which scores likelihood of diabetes based on family history
age	Age in years (integers)
class	Class variable with two string: ['tested_negative', 'tested_positive']

Table of slides 4.5 steps to obtain two “AI Machine Learning predictive Models” for a “diabetes dataset,” one as “Classifier Model” and the other as “Regression Model” using the following MATLAB applications: “Classification Learner” and “Regression Learner” that are included in MATLAB: “Statistics and Machine Learning Toolbox.”

Description

Screen figure

- 1 Open your MATLAB software. In the command prompt enter the command: `>> “classificationLearner”`
 In the “Classification Learner,” click on the “New Session icon” and click submenu “From File.” In the window “Select a file to open” go to the subdirectory downloaded from the book companion website, select “csv” as the file type, then select the file “diabetes.csv” and click on the button “Open”

1. Open your MATLAB software. In the command prompt enter the command: `>> classificationLearner`

In the “Classification Learner”, click on the “New Session icon” and click sub-menu “From File”. In the window “Select a file to open” go to the sub-directory downloaded from the book companion website, select “csv” as the file type, then select the file “diabetes.csv” and click on the button “Open”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

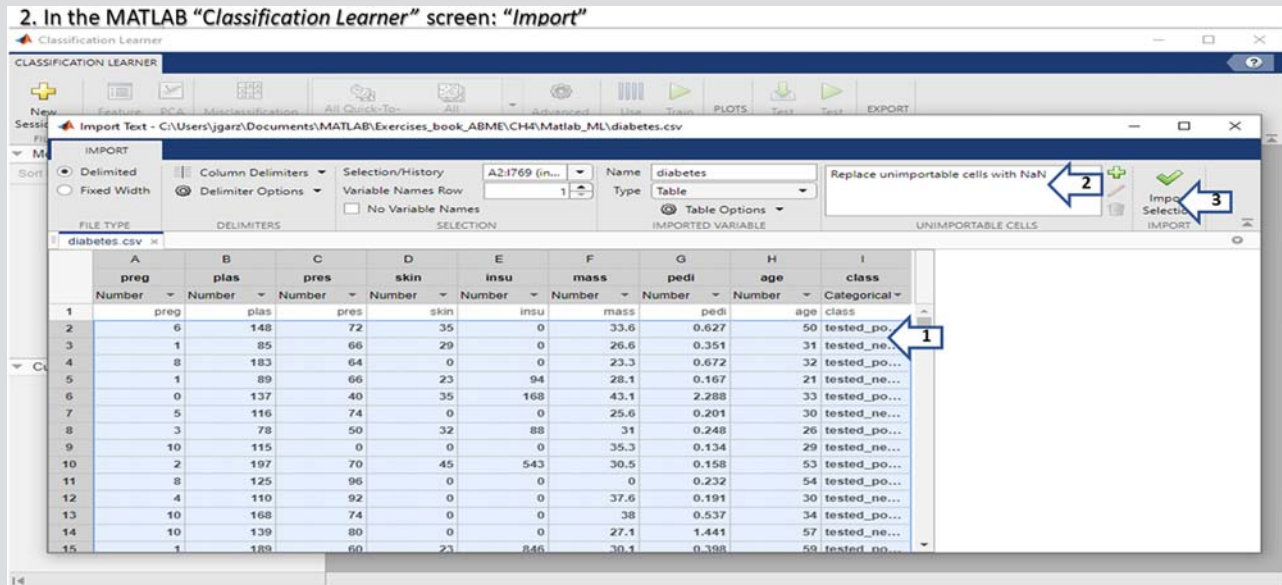
(Continued)

(Continued)

Description

Screen figure

- 2 In the MATLAB "Classification Learner" screen: "Import"
Observe the field or attributes from the dataset "diabetes.csv,"
select "Replace unimportable cells with NaN" and click in
"Import Selection" icon



Observe the field or attributes from the dataset "diabetes.csv", select "Replace unimportable cells with NaN" and click in "Import Selection" icon.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

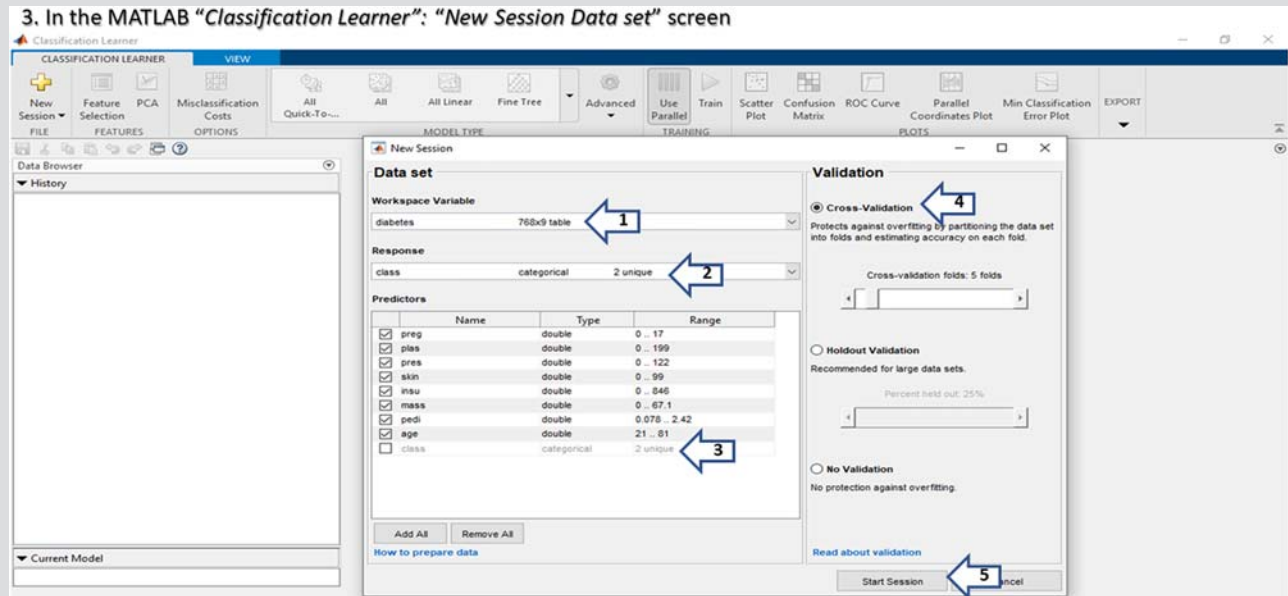
(Continued)

(Continued)

Description

Screen figure

- 3 In the MATLAB “Classification Learner”: “New Session Data set” screen
Observe that the dataset “diabetes.csv” has 789 records with 9 fields or categories, verify “response” that the target is “class” is categorical with 2 unique labels, for validation select “Cross-Validation with 5 folds” and click in the button “Start Session”



Observe that the dataset “diabetes.csv” has 789 records with 9 fields or categories , verify “response” that the target is “class” is categorical with 2 unique labels, for validation select “Cross-Validation with 5 folds” and click in the button “Start Session”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

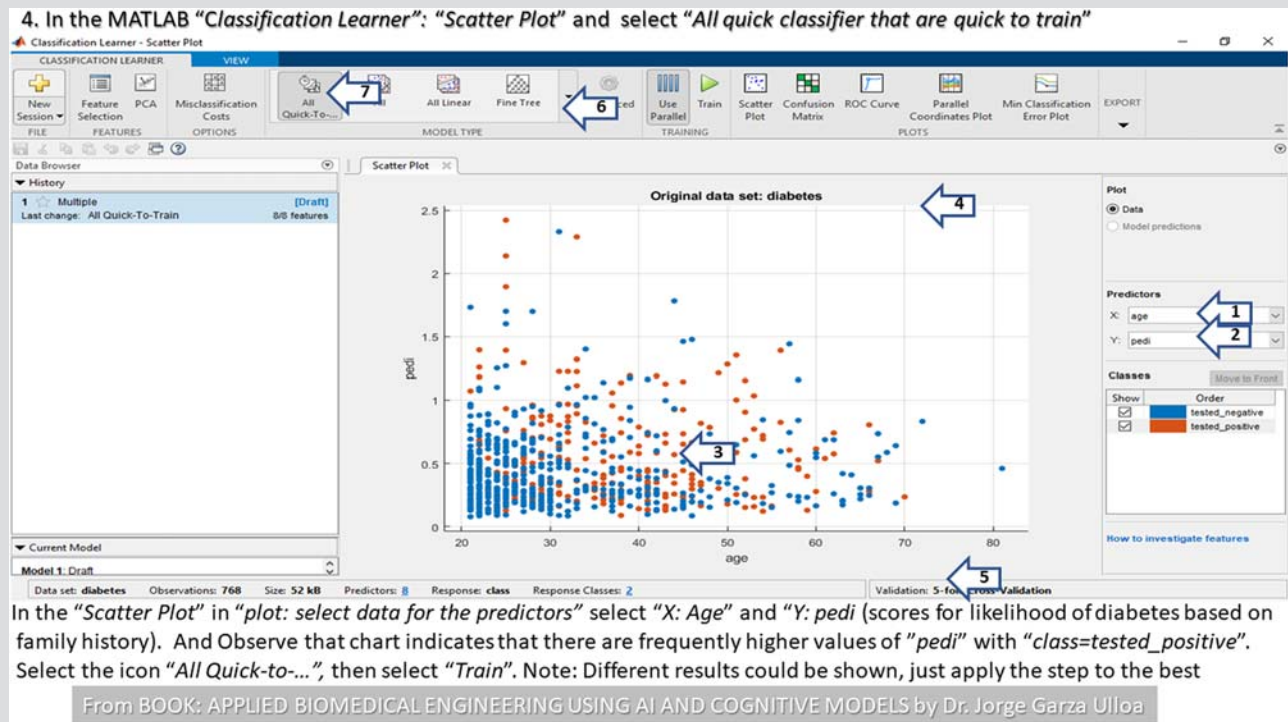
(Continued)

(Continued)

Description

Screen figure

- 4 In the MATLAB “Classification Learner”: “Scatter Plot” and select “All quick classifier that are quick to train”
In the “Scatter Plot” in “plot: select data for the predictors” select “X: Age” and “Y: pedi (scores for likelihood of diabetes based on family history). And Observe that chart indicates that there are frequently higher values of “pedi” with “class = tested_positive.” Select the icon “All Quick-to-...,” then select “Train”. Note: Different results could be shown, just apply the step to the best



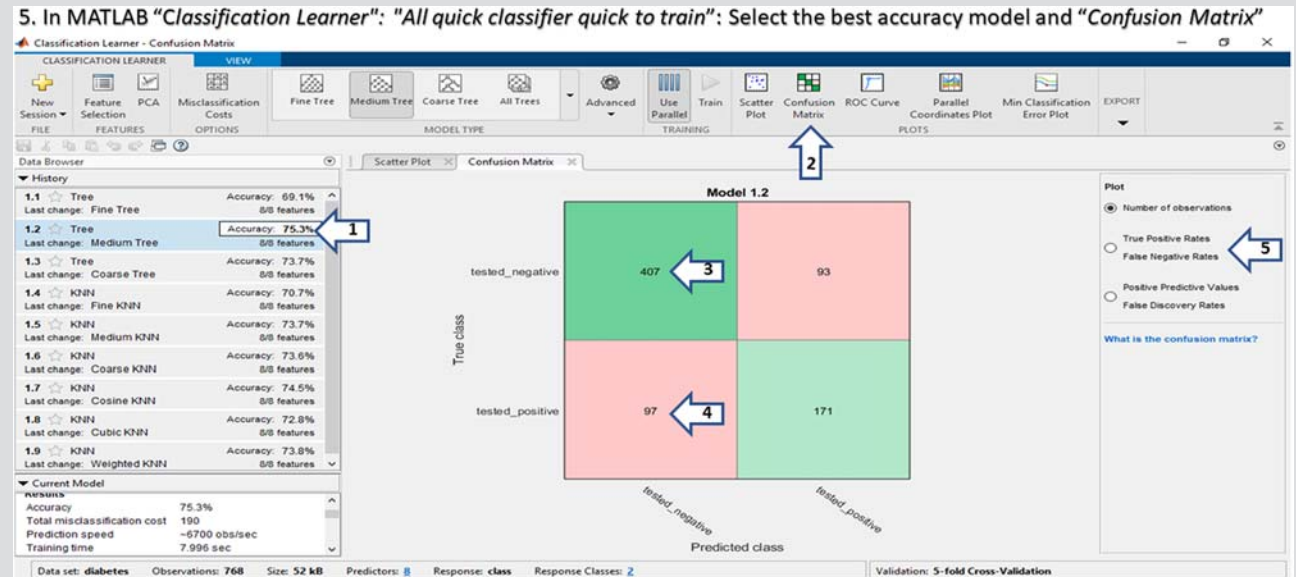
(Continued)

(Continued)

Description

Screen figure

- 5 In MATLAB "Classification Learner": "All quick classifier quick to train": Select the best accuracy model and "Confusion Matrix"
- Select the best model the "Tree-Medium Tree" with "Accuracy of 75.3%," select "Confusion Matrix" icon with option of "Number of observations" that indicate that 500 record with "class" = "tested_negative": 407 were classified correctly, and 93 incorrectly. For "class" = "tested_positive" with 268 record: 171 were classify correctly and 97 incorrectly."
- Select the option "True positive rates/False negative rates".
- Note: Different results could be shown



Select the best model the "Tree- Medium Tree" with "Accuracy of 75.3%", select "Confusion Matrix" icon with option of "Number of observations" that indicate that 500 record with "class"="tested_negative": 407 were classified correctly, and 93 incorrectly. For "class"="tested_positive" with 268 record: 171 were classify correctly and 97 incorrectly". Select the option "True positive rates/ False negative rates".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

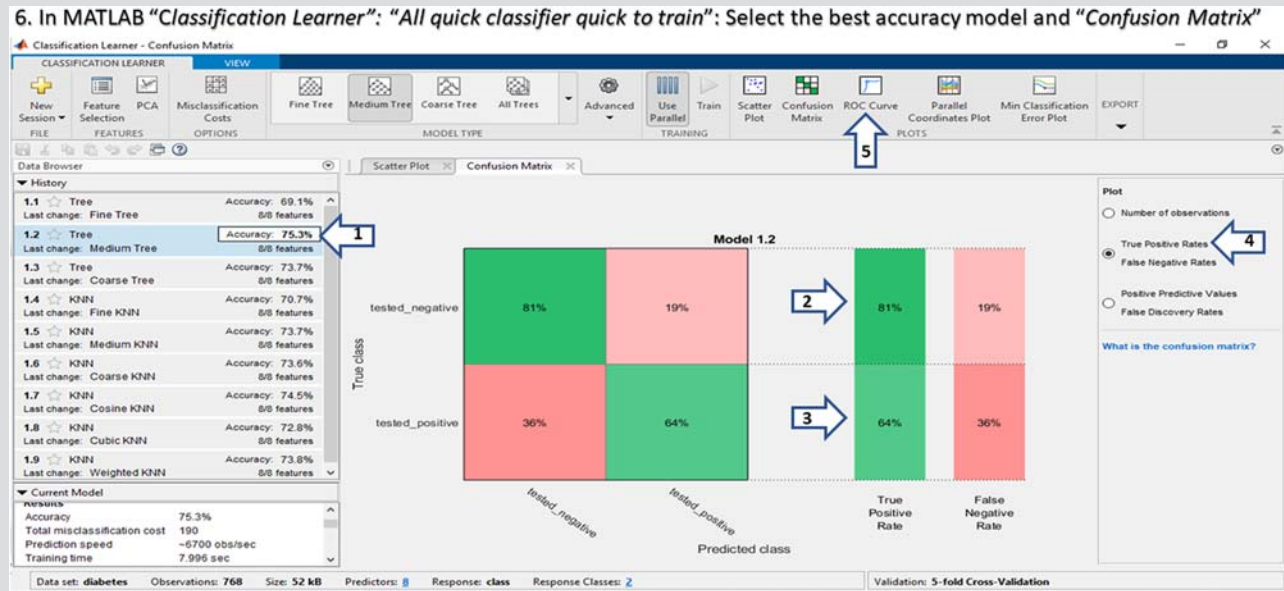
(Continued)

(Continued)

Description

Screen figure

- 6 In MATLAB "Classification Learner": "All quick classifier quick to train": Select the best accuracy model and "Confusion Matrix"
"Confusion Matrix"
Select the best model: "Tree-Medium Tree" with "Accuracy of 75.3%," select "Confusion Matrix" icon with option of "True positive rates/False negative rates," it indicate that 500 record with "class" = "tested_negative": 81% were classified correctly and 19% incorrectly. For "class" = "tested_positive" with 268 record: 64% were classify correctly and 36% incorrectly." Select the option "ROC Curve".
Note: Different results could be shown



Select the best model: "Tree- Medium Tree" with "Accuracy of 75.3%," select "Confusion Matrix" icon with option of "True positive rates/False negative rates," it indicate that 500 record with "class"="tested_negative": 81% were classified correctly and 19% incorrectly. For "class"="tested_positive" with 268 record: 64% were classify correctly and 36% incorrectly". Select the option "ROC Curve"
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Description

Screen figure

7 In MATLAB "Classification Learner": "All quick classifier quick to train": Select the best accuracy model and "ROC Curve"
"Curve"
Select the best model: "Tree-Medium Tree" with "Accuracy of 76%," select "ROC Curve" icon that shows an AUC = 0.77, with the option of "Plot" "Positive class with tested_negative" where the current classifier is on "(0.36,0.81)." Select "Positive class with tested_positive" where the current classifier is on "(0.19,0.64)." Note: ROC curve shows a low true positive rate versus false positive rate for the currently selected trained classifier

7. In MATLAB "Classification Learner": "All quick classifier quick to train": Select the best accuracy model and "ROC Curve"

History	Accuracy	Features
1.1 Tree	69.1%	88
1.2 Tree	75.3%	88
1.3 Tree	73.7%	88
1.4 KNN	70.7%	88
1.5 KNN	73.7%	88
1.6 KNN	73.6%	88
1.7 KNN	74.5%	88
1.8 KNN	72.8%	88
1.9 KNN	73.8%	88

Current Model Summary:

- Accuracy: 75.3%
- Total misclassification cost: 190
- Prediction speed: ~6700 obs/sec
- Training time: 7.996 sec

ROC Curve Data:

Plot	Positive class	Negative classes	Current Classifier (x, y)	AUC
Left	tested_negative	tested_positive	(0.36, 0.81)	0.77
Right	tested_positive	tested_negative	(0.19, 0.64)	0.77

Select the best model: "Tree-Medium Tree" with "Accuracy of 76%," select "ROC Curve" icon that shows an AUC=0.77, with the option of "Plot" "Positive class with tested_negative" where the current classifier is on "(0.36,0.81)". Select "Positive class with tested_positive" where the current classifier is on "(0.19,0.64)". Note: ROC curve shows a low true positive rate versus false positive rate for the currently selected trained classifier

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

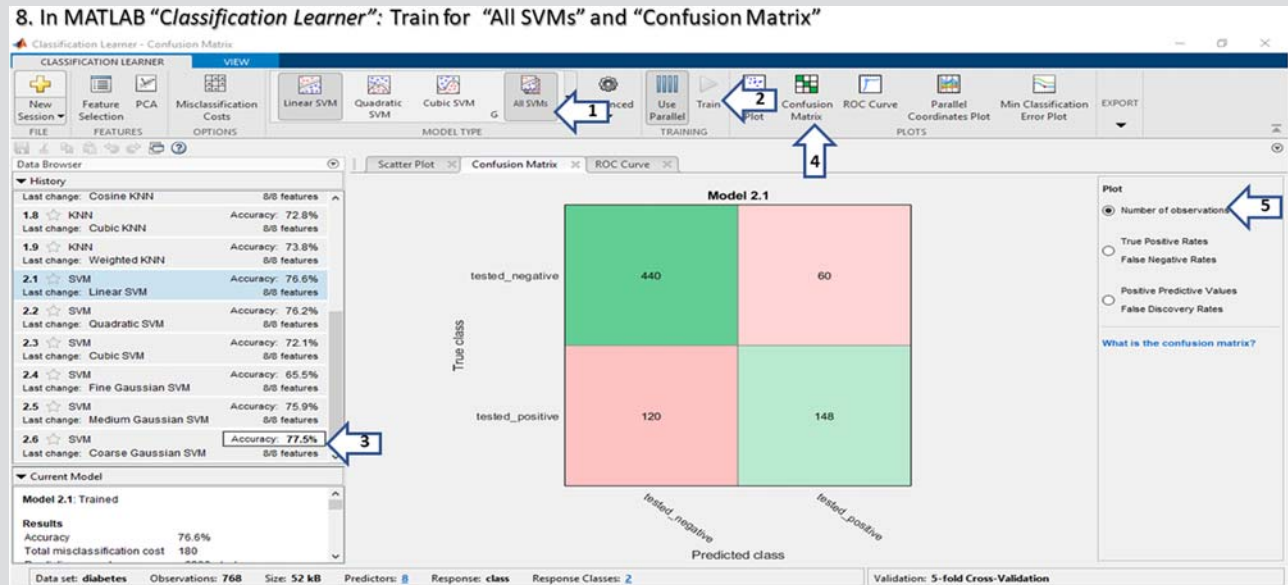
(Continued)

(Continued)

Description

Screen figure

- 8 In MATLAB “Classification Learner”: Train for “All SVMs” and “Confusion Matrix”
Select “All SVMs” and click on “Train,” then select the best model: “SVM Linear SVM” with “Accuracy of 77.5%,” select “Confusion Matrix” with the option of “Number of observations” that indicate that 500 records with “class” = “tested_negative”: 440 were classified correctly and 60 incorrectly. But for “class” = “tested_positive” with 268 record: 148 were classify correctly and 120 incorrectly”



Select “All SVMs” and click on “Train,” then select the best model : “SVM Linear SVM” with “Accuracy of 77.5%”, select “Confusion Matrix” with the option of “Number of observations” that indicate that 500 records with “class”=“tested_negative”: 440 were classified correctly and 60 incorrectly. But for “class”=“tested_positive” with 268 record: 148 were classify correctly and 120 incorrectly”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

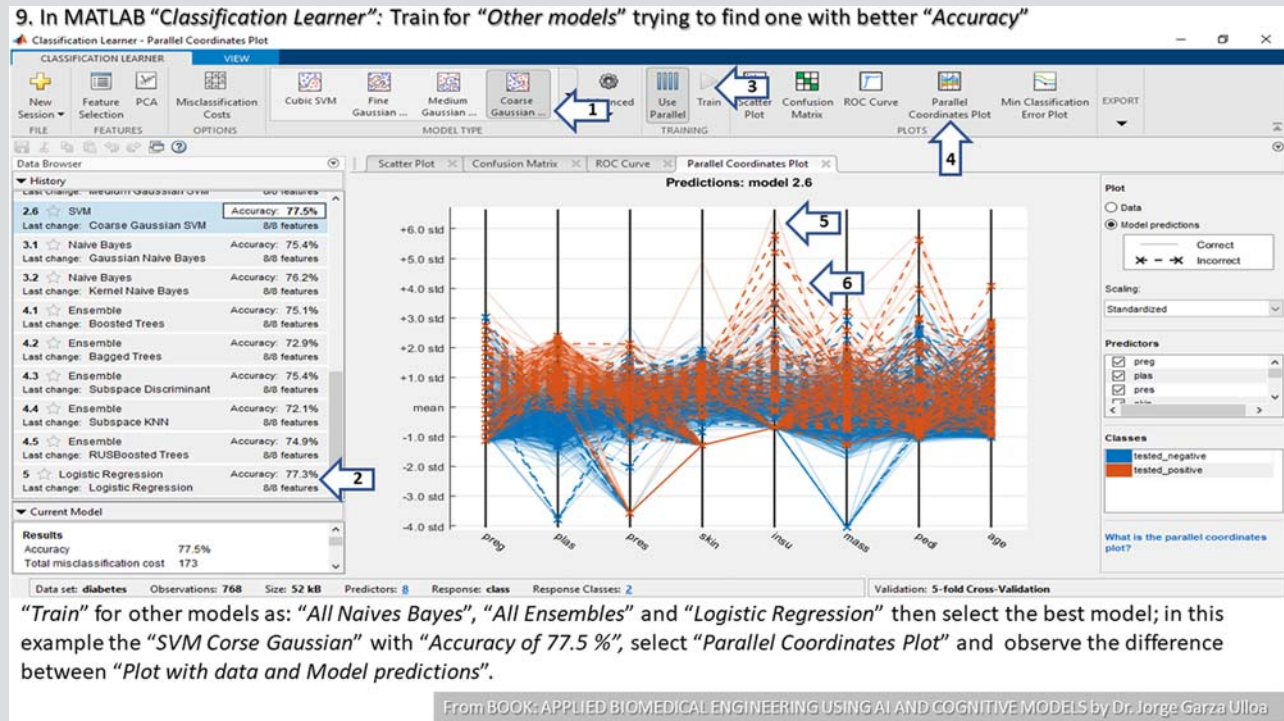
(Continued)

(Continued)

Description

Screen figure

- 9 In MATLAB *Classification Learner*: Train for *Other models* trying to find one with better *Accuracy* *Train* for other models as: *All Naives Bayes*, *All Ensembles*, and *Logistic Regression* then select the best model; in this example the *SVM Corse Gaussian* with *Accuracy of 77.5%*, select *Parallel Coordinates Plot* and observe the difference between *Plot with data and Model predictions*



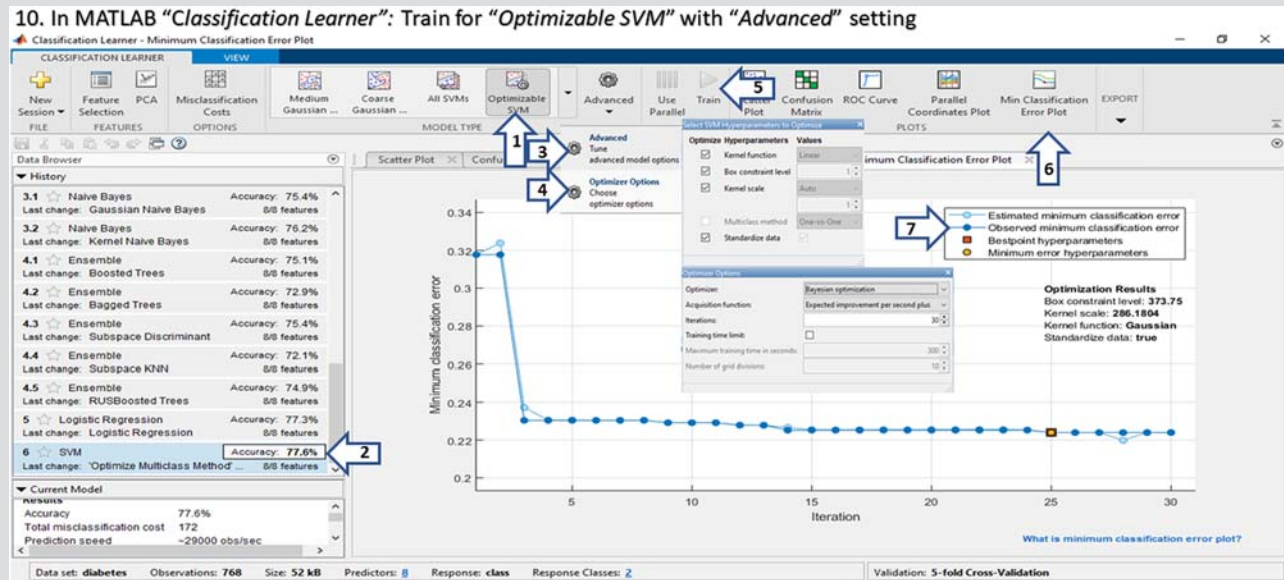
(Continued)

(Continued)

Description

Screen figure

10 In MATLAB "Classification Learner": Train for "Optimizable SVM" with "Advanced" setting
Select the best model: in this example the "Optimizable SVM" with "Accuracy of 77.6%," select "Advanced" icon and "Advanced Tune" to see the "Optimize Hyperparameters," then select "Optimizer Options" as shown. "Train" model and generate a "Min Classification Error Plot" with 30 iterations that indicates the best value for the "Observed minimum classification error"



Select the best model in this example the "Optimizable SVM" with "Accuracy of 77.6 %", select "Advanced" icon and "Advanced Tune" to see the "Optimize Hyperparameters," then select "Optimizer Options" as shown. "Train" model and generate a "Min Classification Error Plot" with 30 iterations that indicates the best value for the "Observed minimum classification error".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

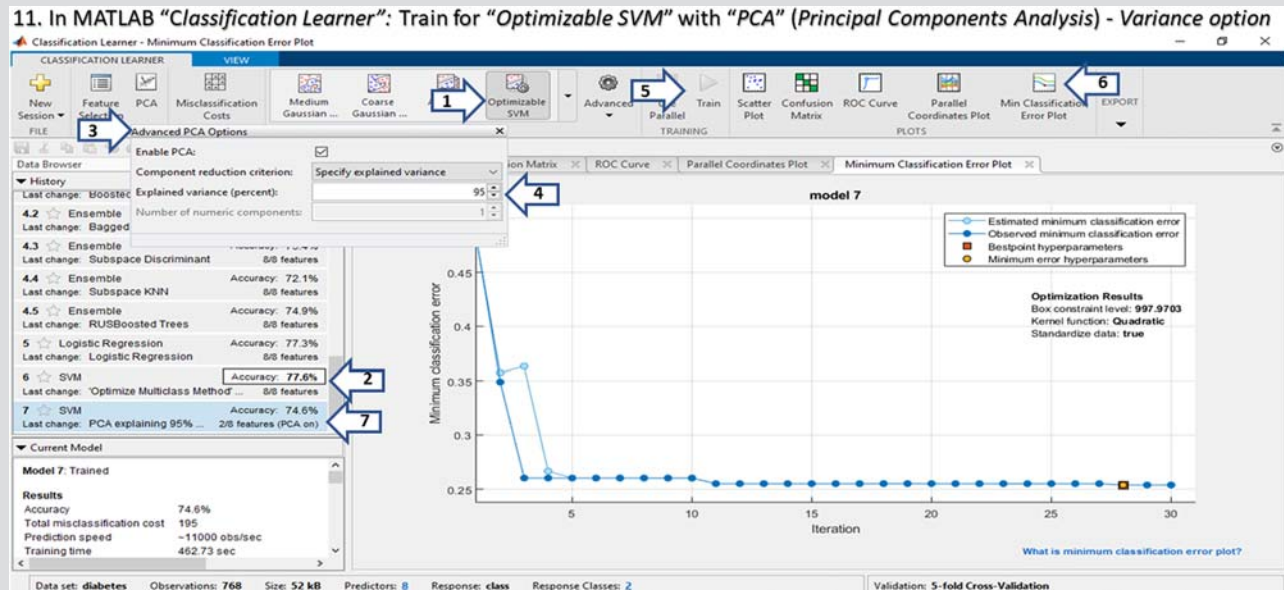
(Continued)

(Continued)

Description

Screen figure

11 In MATLAB “Classification Learner”: Train for “Optimizable SVM” with “PCA” (Principal Components Analysis) - Variance option. Select the best model in this example is “Optimizable SVM” with “Accuracy of 77.6%,” select “PCA” icon to “Enable PCA,” with “Accuracy of 77.6%,” select “PCA” icon to “Enable PCA,” with a “Component reduction criterion”: “Specify explained variance of 95%.” “Train” model and generate a “Min Classification Error Plot” with 30 iterations that indicates the best value for the “Observed minimum classification error,” and “lower Accuracy of 74.6%”



Select the best model in this example is “Optimizable SVM” with “Accuracy of 77.6 %”, select “PCA” icon to “Enable PCA”, with a “Component reduction criterion”: “Specify explained variance of 95%”. “Train” model and generate a “Min Classification Error Plot” with 30 iterations that indicates the best value for the “Observed minimum classification error”, and “lower Accuracy of 74.6%”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

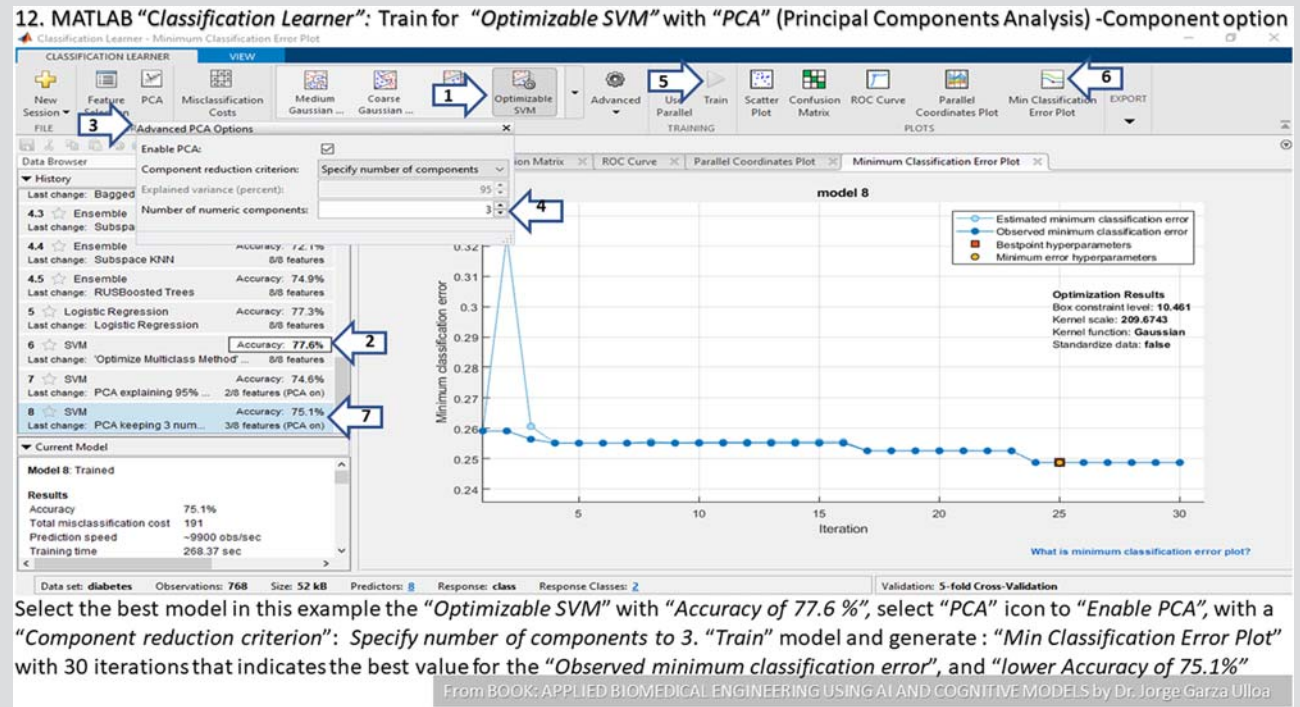
(Continued)

(Continued)

Description

Screen figure

12 MATLAB "Classification Learner": Train for "Optimizable SVM" with "PCA" (Principal Components Analysis) - Component option Select the best model in this example the "Optimizable SVM" with "Accuracy of 77.6%," select "PCA" icon to "Enable PCA," with a "Component reduction criterion": Specify number of components to 3. "Train" model and generate: "Min Classification Error Plot" with 30 iterations that indicates the best value for the "Observed minimum classification error," and "lower Accuracy of 75.1%"



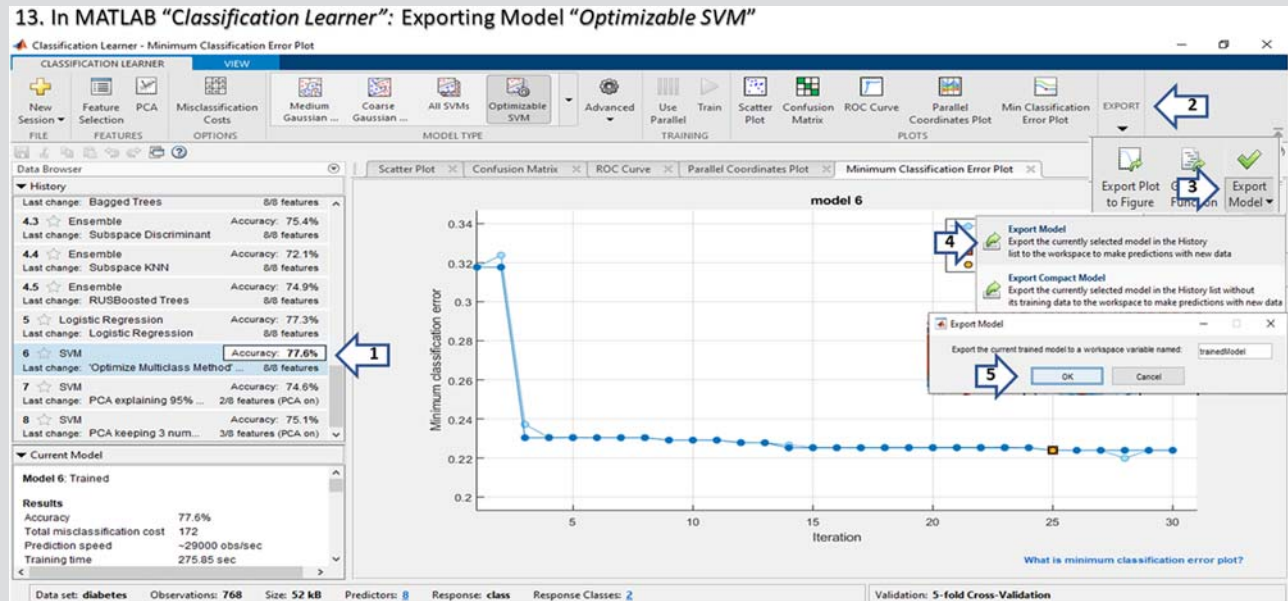
(Continued)

(Continued)

Description

Screen figure

- 13 In MATLAB "Classification Learner": Exporting Model "Optimizable SVM"
Select the best model in this example the "Optimizable SVM" with "Accuracy of 77.6%," click on the "Export" icon, then in "Export Model," accept the suggested workspace variable named: "trainedModel" clicking on the "OK button"



Select the best model in this example the "Optimizable SVM" with "Accuracy of 77.6 %", click on the "Export" icon, then in "Export Model", accept the suggested workspace variable named: "trainedModel" clicking on the "OK button"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Description

Screen figure

- 14 In MATLAB workplace main screen test the trained model: "Optimized SVM"
In the "MATLAB workspace" verify that the exported "trainedModel" is available, enter the instruction: "T = readtable('diabetes_test.csv')"
click on the variable "T" to see the testing values, then enter the instruction: "yfit = trainedModel.predictFcn(T)"
and compare the results obtained with the expected values. Close "Classification Learner"

14. In MATLAB workplace main screen test the trained model: "Optimized SVM"

1	2	3	4	5	6	7	8	9	10	11	12	13
preg	plas	pres	skin	insu	mass	pedi	age	class				
10	30	0	0	0	35.3000	0.0500	29	'tested_negative'				
2	197	70	45	543	30.5000	0.1580	53	'tested_positive'				
7	147	76	0	0	39.4000	0.2570	43	'tested_positive'				
1	97	66	15	140	23.2000	0.4870	22	'tested_negative'				

```
Command Window
To make predictions on a new table, T:
yfit = trainedModel.predictFcn(T)
For more information, see How to predict using an exported model.
>> T=readtable('diabetes_test.csv');
>> yfit = trainedModel.predictFcn(T)

yfit =

4x1 categorical array

tested_negative
tested_positive
tested_positive
tested_negative
```

In the "MATLAB workspace" verify that the exported "trainedModel" is available, enter the instruction :
"T=readtable('diabetes_test.csv')", click On the variable "T" to see the testing values, then enter the instruction:
"yfit = trainedModel.predictFcn(T)", and compare the results obtained with the expected values. Close "Classification Learner"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

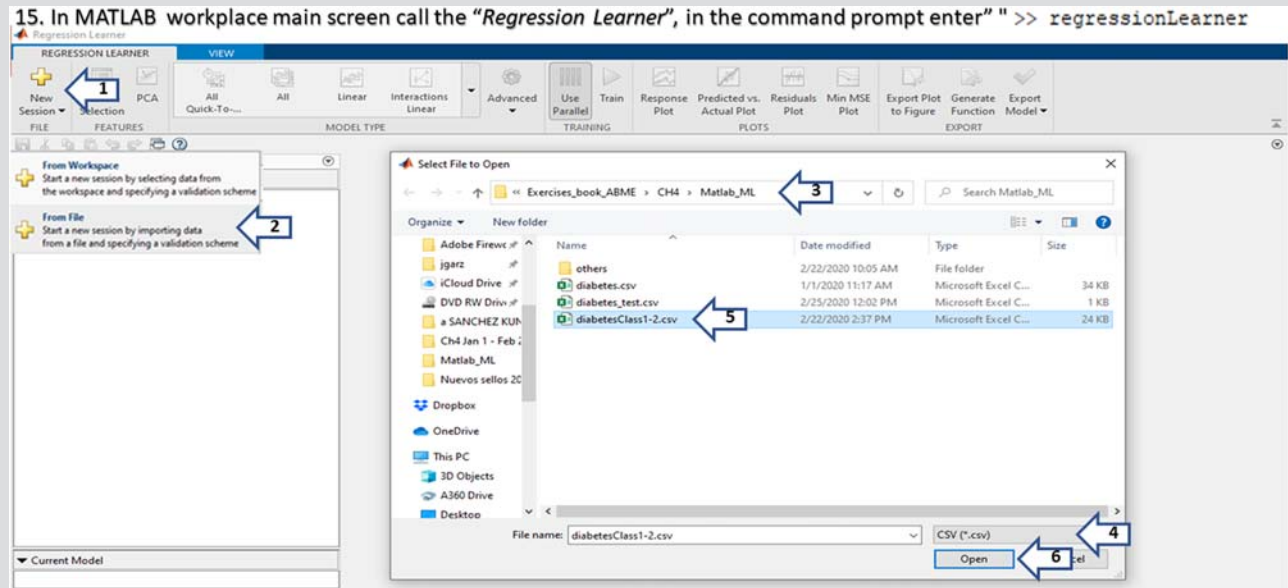
(Continued)

(Continued)

Description

Screen figure

- 15 In MATLAB workplace main screen call the "Regression Learner," in the command prompt enter " >> regressionLearner"
"regressionLearner"
In the "Regression Learner," click on the "New Session icon" and click on "From File."
In the window for "Select file to Open" go to the subdirectory downloaded from the book companion website, select "csv," as the file type, then select "diabetesClass1-2.csv" and click on the button "Open"



In the "Regression Learner", click on the "New Session icon" and click on "From File". In the window for "Select file to Open" go to the sub-directory downloaded from the book companion website, select "csv," as the file type, then select "diabetesClass1-2.csv" and click on the button "Open"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

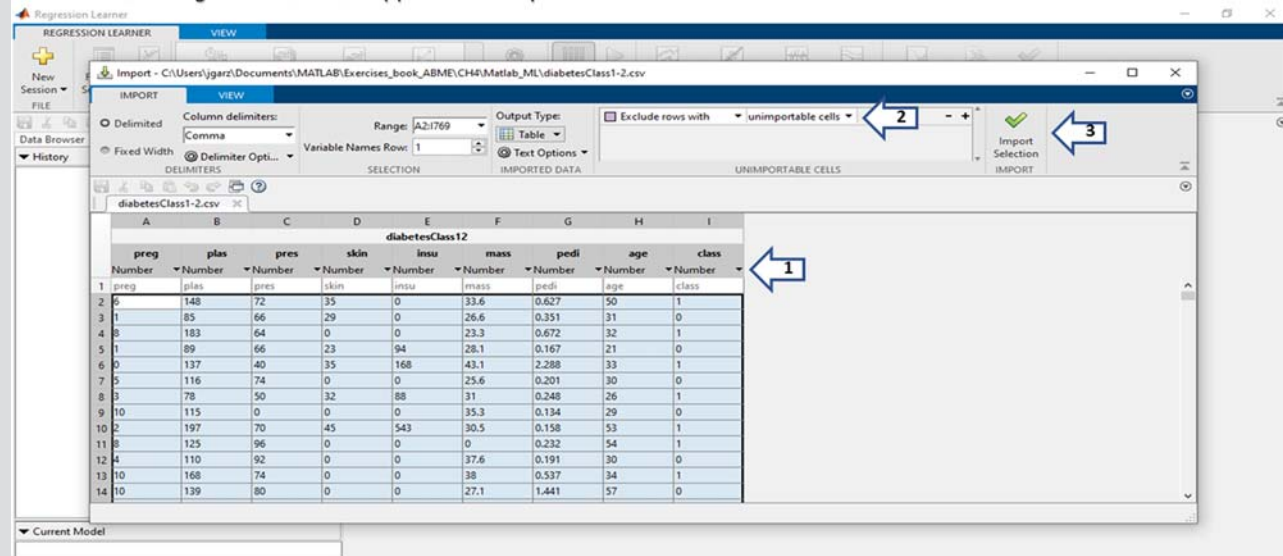
(Continued)

Description

Screen figure

- 16 In MATLAB “Regression Learner” application: “Import” screen
Observe the importation of the dataset “diabetesClass1–2.csv,” its contains the same data that “diabetes.csv” with the exception that the attribute ‘class’ is now numeric: 1 means ‘tested_positive’ and 0 means ‘tested_negative.’ Select in “Importable Cells” section the option of “Exclude row with” “unimportable cells” and click in “Import Selection” icon

16. In MATLAB “Regression Learner” application: “Import” screen



Observe the importation of the dataset “diabetesClass1-2.csv,” its contains the same data that “diabetes.csv” with the exception that the attribute ‘class’ is now numeric: 1 means ‘tested_positive’ and 0 means ‘tested_negative.’ Select in “Importable Cells” section the option of “Exclude row with” “unimportable cells” and click in “Import Selection” icon

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

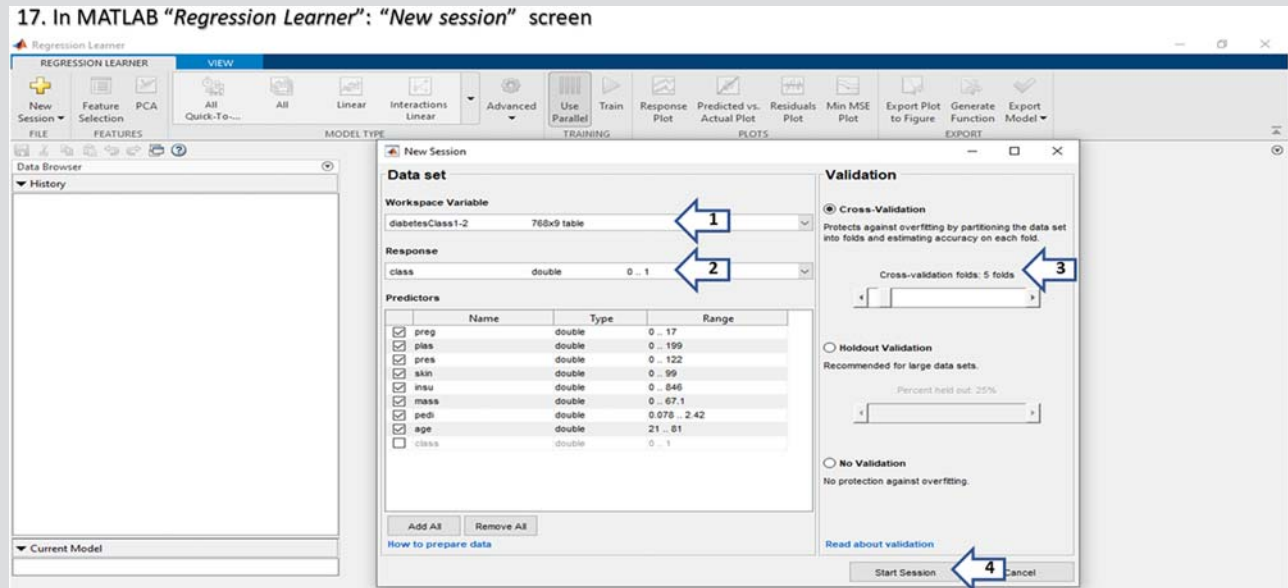
(Continued)

(Continued)

Description

Screen figure

- 17 In MATLAB "Regression Learner": "New session" screen
Observe that the dataset "diabetesClass1-2.csv" has 789 records with 9 fields or attributes, verify in "response" that the target is "class" as double, for validation select "Cross-Validation 5 folds" and click in the button "Start Session"



Observe that the dataset "diabetesClass1-2.csv" has 789 records with 9 fields or attributes, verify in "response" that the target is "class" as double, for validation select "Cross-Validation 5 folds" and click in the button "Start Session"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

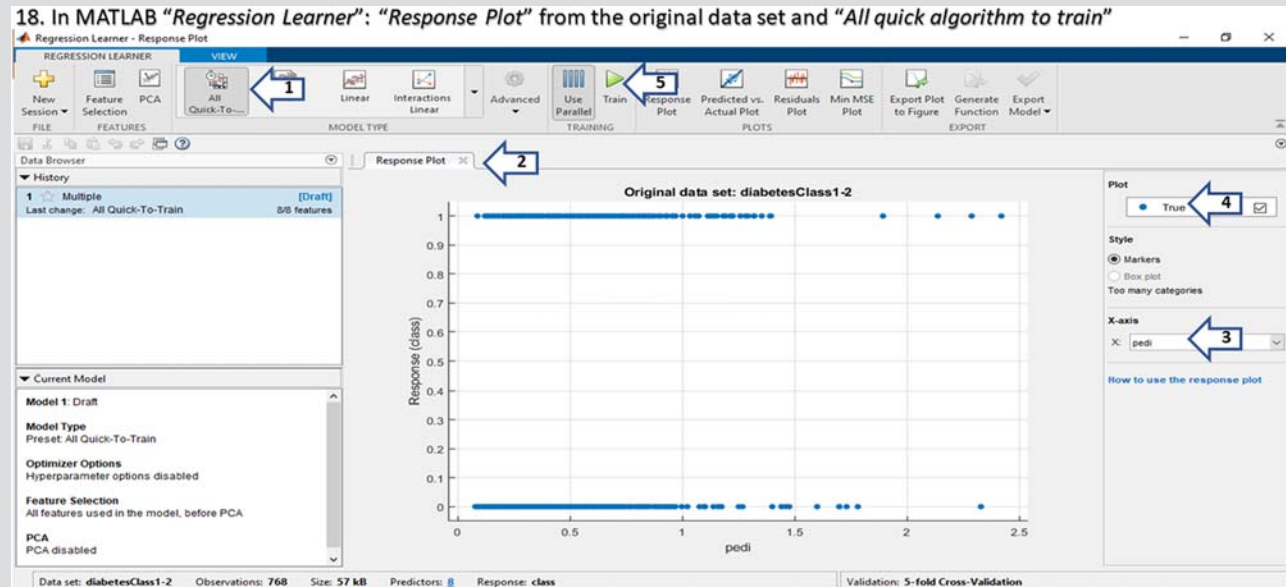
(Continued)

(Continued)

Description

Screen figure

- 18 In MATLAB "Regression Learner": "Response Plot" from the original dataset and "All quick algorithm to train" Select "All quick algorithm to train," in the "Response Plot" from the original dataset select x-axis: "pedi" and observe that there are more "tested_positive = 1" with higher "pedi" values, then select "Train"



Select "All quick algorithm to train", in the "Response Plot" from the original data set select x-axis: "pedi" and observe that there are more "tested_positive=1" with higher "pedi" values, then select "Train"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

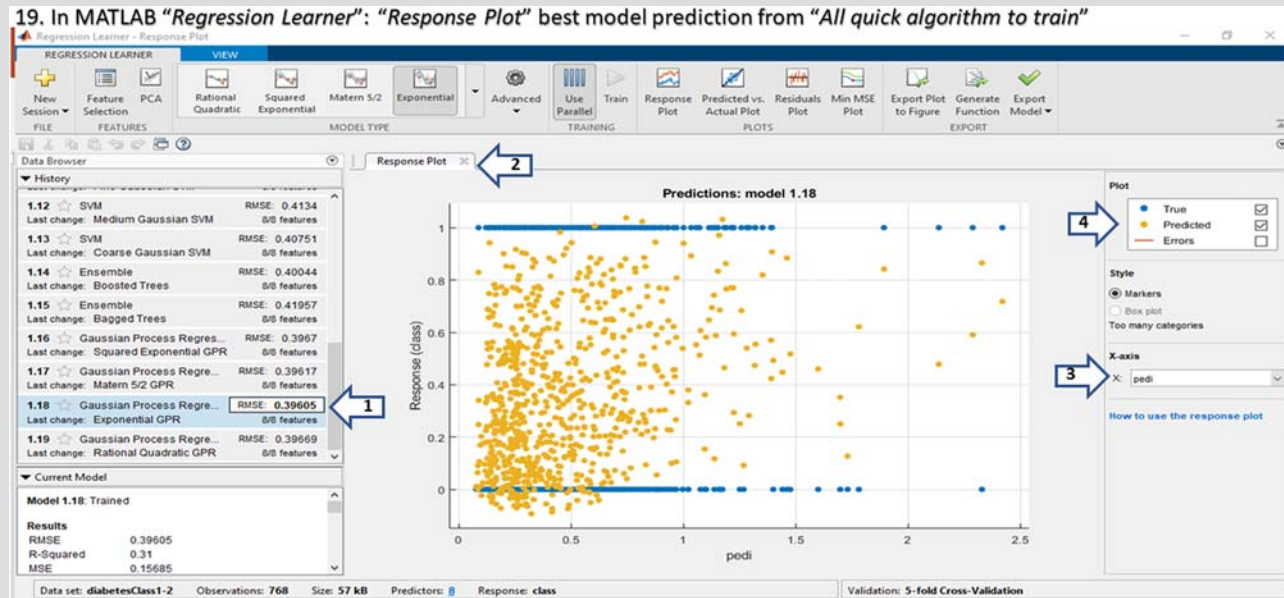
(Continued)

(Continued)

Description

Screen figure

19 In MATLAB "Regression Learner": "Response Plot" model prediction from "All quick algorithm to train" Select the best mode: "Gaussian Process Regression" with an "RMSE = 0.39605" (Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors), the lower is the best). Observe the "Response Plot" for the prediction value of 'pedi' versus 'class'



Select the best mode: "Gaussian Process Regression" with a "RMSE=0.39605" (Root Mean Square Error (RMSE) is the standard deviation of the residuals (prediction errors), the lower is the best). Observe the "Response Plot" for the prediction value of 'pedi' vs 'class'

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

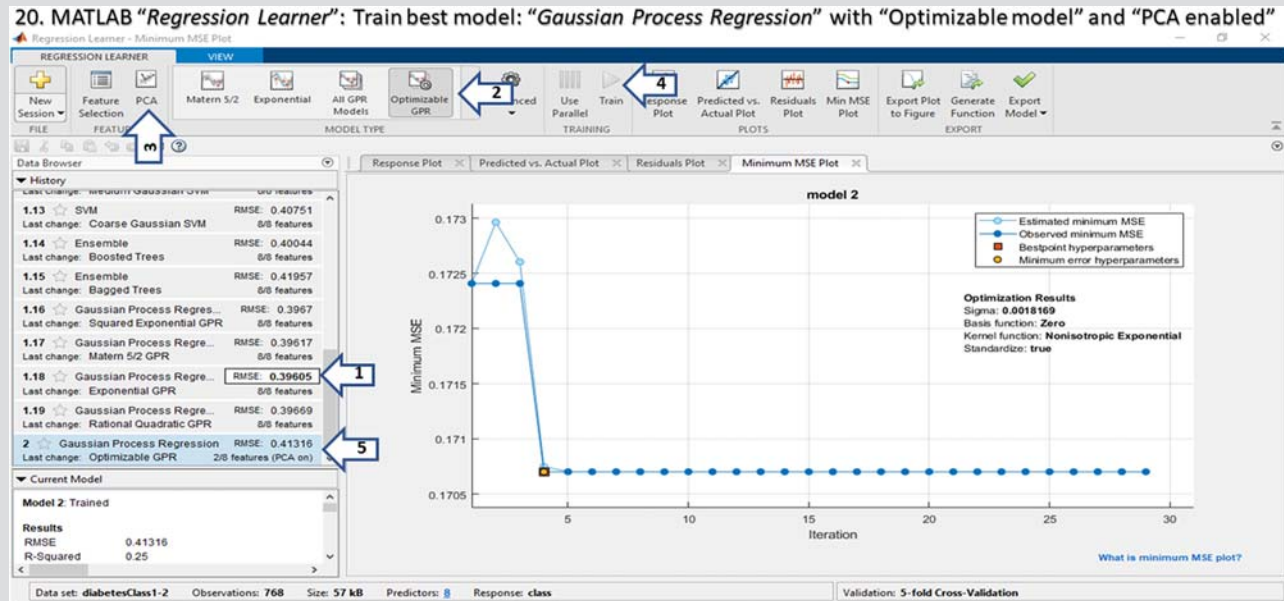
(Continued)

(Continued)

Description

Screen figure

20 MATLAB "Regression Learner": Train best model: "Gaussian Process Regression" with "Optimizable model" and "PCA enabled" Select the best mode: "Gaussian Process Regression" with an "RMSE = 0.39605," enable "PCA," select "Optimizable GPR" and "Train" with these parameters, a "Minimum MSE Plot" is generated and the "RMSE is 0.41316"



Select the best mode: "Gaussian Process Regression" with a "RMSE=0.39605", enable "PCA", select "Optimizable GPR" and "Train" with these parameters, a "Minimum MSE Plot" is generated and the "RMSE is 0.41316".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

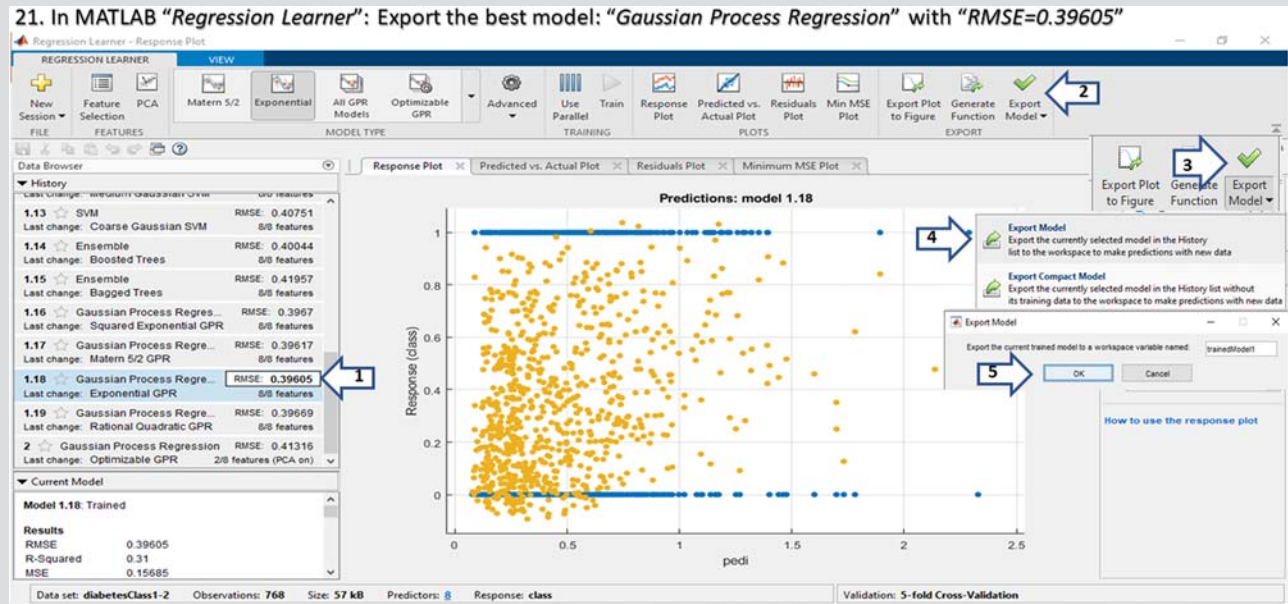
(Continued)

(Continued)

Description

Screen figure

21 In MATLAB "Regression Learner": Export the best model: "Gaussian Process Regression" with "RMSE = 0.39605" Select the best mode: "Gaussian Process Regression" with an "RMSE = 0.39605," click on "Export Model," then "Export Model" and accept the exportation of a structure carriable of trained model as "trainedModel1"



Select the best mode: "Gaussian Process Regression" with a "RMSE=0.39605", click on "Export Model", then "Export Model" and accept the exportation of a structure carriable of trained model as "trainedModel1"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

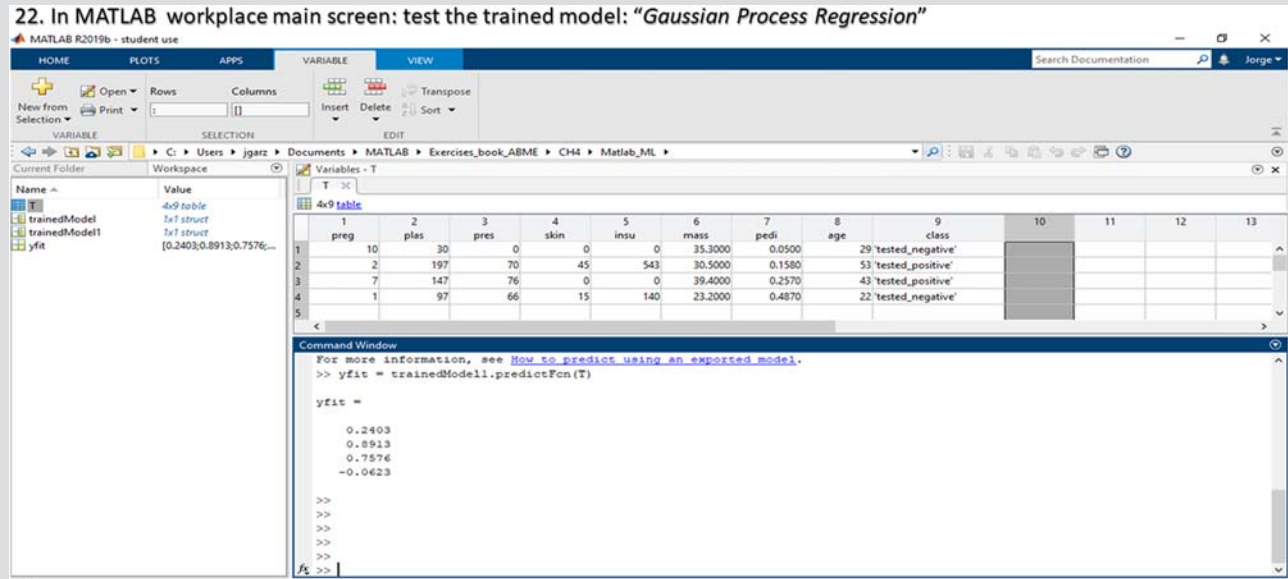
(Continued)

(Continued)

Description

Screen figure

22 In MATLAB workplace main screen: test the trained model: "Gaussian Process Regression"
In the MATLAB workspace: verify that the "trainedModel1" is available, the variable "T" has the testing values, then enter the instruction: "yfit = trainedModel1.predictFcn(T)" and compare the results obtained with the expected values
Note: if "class" is less or equal than 0.5 then it is 'tested_negative' else then is 'tested_positive.' Save the Workspace as: "trainedModels" and close MATLAB and the "Regression Learner"



In the MATLAB workspace: verify that the "trainedModel1" is available, the variable "T" has the testing values, then enter the instruction: "yfit = trainedModel1.predictFcn(T)", and compare the results obtained with the expected values. Note: if "class" is less or equal than .5 then it is 'tested_negative' else then is 'tested_positive'. Save the Workspace as: "trainedModels" and close MATLAB and the "Regression Learner".

Conclusions

We can have the following conclusions for obtaining a “ML Classifier models” [32] and a “ML Regression models” using the following MATLAB applications: “Classification Learner” and “Regression Learner,” which are included in MATLAB: “Statistics and Machine Learning Toolbox”:

- “ML Classifier models” using MATLAB “Classification Learner” allows the import of a dataset from “files” or “MATLAB workspace,” specifying the validation type and the response variable. We can select a training as “All quick classifier that are faster to train,” then train for other models’ types: “Decision Tress Models,” “Discriminant Analysis Models,” “Logistic Regression Classifiers,” “Naives Bayes Classifier Models,” “Support Vector Machine Classifier Models,” “Nearest Neighbor Classifier Models,” and “Ensemble Classifiers Models.” Specifying: “feature selection,” “PCA,” and “optimization of classifier algorithms.” Obtaining: “Scatter Plots,” “Confusion Matrix,” “ROC Curves,” “Parallel Coordinates Plot, etc. It also allows export the best model as a “MATLAB function” or a “workspace variable” to predict new values for the dataset. For this “diabetes dataset” the best “ML Classifier model” was the “Optimizable SVM” with “Accuracy of 77.6%,” it works well to predict the dataset for testing.
- “ML Regression models” using MATLAB “Regression Learner” allow the import of datasets from “files” or “MATLAB workspace,” specifying the validation type and the response variable that must be numeric. We can select a training as “All quick regression that are faster to train,” then train for other models’ types as: “Regression Trees models,” “Linear Regression models,” “SVM regression models,” “Gaussian Process Regression Models” and “Ensembles of Trees.” Specifying: “feature selection,” “PCA,” and “optimization of classifier algorithms.” Obtaining: “Response Plots,” “Predicted versus Actual Plot,” “Residuals Plots,” “Min MSE Plot,” etc. It allows “export” of the best model as a “MATLAB function” or a “workspace variable” to predict new values for the dataset. For this “diabetes dataset” the best “ML Regression model” was the “Gaussian Process Regression” with “RMSE = 0.39605”%.

Recommendations

- Update the dataset or create a new one adding the following diabetes attributes: “urine test” and “hemoglobin A1C test,” as explained in the section “Diabetes tests” of this research.
- A larger dataset will help to analyze diabetes with more precision.

References

- [1] E.O. Neftci, B.B. Averbeck, Reinforcement learning in artificial and biological systems, *Nat. Mach. Intell.* 1 (2019) 133–143. Available from: <https://doi.org/10.1038/s42256-019-0025-4>.

- [2] J.P. Pfister, T. Toyozumi, D. Barber, W. Gerstner, Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning, *Neural Comput.* 18 (2006) 1318–1348.
- [3] M.K. Benna, S. Fusi, Computational principles of biological memory. Preprint at <https://arxiv.org/abs/1507.07580>, 2015.
- [4] H.F. Harlow, The formation of learning sets, *Psychol. Rev.* 56 (1949) 51–65.
- [5] C.K. Reddy, Y. Li, A review of clinical prediction models, in: C. K. Reddy, C.C. Aggarwal (Eds.), *Healthcare Data Analytics*, Chapman and Hall/CRC Press, 2015.
- [6] L. Yang, K. Pelckmans, Member, IACSIT, machine learning approaches to survival analysis: case studies in microarray for breast cancer, *Int. J. Mach. Learn. Comput.* 4 (6) (2014).
- [7] J. Silva, N. Varela, L.A.B. López, R.H.R. Millán, Association rules extraction for customer segmentation in the SMEs sector using the Apriori algorithm, *Procedia Computer Sci.* 151 (2019) 1207–1212. Available from: <https://doi.org/10.1016/j.procs.2019.04.173>. ISSN 1877-0509.
- [8] N. Zeng, H. Xiao, Inferring implications in semantic maps via the Apriori algorithm, *Lingua* (2020) 102808. Available from: <https://doi.org/10.1016/j.lingua.2020.102808>. ISSN 0024-3841.
- [9] https://en.wikipedia.org/wiki/Naive_Bayes_classifier (accessed 28.02.20).
- [10] <https://towardsdatascience.com/https-medium-com-lorlri-classification-and-regression-analysis-with-decision-trees-c43cdbc58054> (accessed 29.02.20).
- [11] <https://www.analyticsvidhya.com/blog/2016/04/complete-tutorial-tree-based-modeling-scratch-in-python/> (accessed 29.02.20).
- [12] C. Olaru, L. Wehenkel, A complete fuzzy decision tree technique, *Fuzzy Sets Syst.* 138 (2) (2003) 221–254. Available from: [https://doi.org/10.1016/S0165-0114\(03\)00089-7](https://doi.org/10.1016/S0165-0114(03)00089-7) ISSN 0165-0114. Available from: <http://www.sciencedirect.com/science/article/pii/S0165011403000897>.
- [13] Ü. Doğan, T. Glasmachers, C. Igel, “A unified view on multi-class support vector classification” (PDF), *J. Mach. Learn. Res.* 17 (2016) 1–32.
- [14] B. Mohebbi, A. Tahmassebi, A. Meyer-Baese, A.H. Gandomi, *Probabilistic Neural Networks: A Brief Overview of Theory, Implementation, and Application*, Elsevier, 2020, pp. 347–367. 10.1016/B978-0-12-816514-0.00014-X.
- [15] K. Ghazvini, M. Yousefi, F. Firoozeh, S. Mansouri, Predictors of tuberculosis: application of a logistic regression model, *Gene Rep.* 17 (2019) 100527. Available from: <https://doi.org/10.1016/j.genrep.2019.100527>. ISSN 2452-0144.
- [16] S. Belciug, Logistic regression paradigm for training a single-hidden layer feedforward neural network. Application to gene expression datasets for cancer research, *J. Biomed. Inform.* 102 (2020) 103373. Available from: <https://doi.org/10.1016/j.jbi.2019.103373>. ISSN 1532-0464.
- [17] <https://towardsdatascience.com/advanced-ensemble-classifiers-8d7372e74e40> (accessed 28.02.20).
- [18] T.K. Ho, Random decision forests (PDF), in: *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, 14–16 August 1995, Montreal, QC, Canada, 1995, pp. 278–282. Archived from the original (PDF) on 17 April 2016. Retrieved 5 June 2016. [T.K. Ho. The random subspace method for constructing decision forests (PDF). *IEEE Transactions on*

- Pattern Analysis and Machine Intelligence. 20 (8), (1998), 832–844. doi:10.1109/34.709601].
- [19] D. Dua, C. Graff, UCI machine learning repository [mL]. School of Information and Computer Science, California, <https://archive.ics.uci.edu/mL/support/diabetes>, 2019 (accessed 18.02.20).
- [20] <https://www.webmd.com/diabetes/guide/risk-factors-for-diabetes#1> (accessed 26.02.20).
- [21] <https://www.mayoclinic.org/diseases-conditions/heart-disease/symptoms-causes/syc-20353118> (accessed 19.02.20).
- [22] Hungarian Institute of Cardiology, Budapest: Andras Janosi, M. D., University Hospital, Zurich, Switzerland: William Steinbrunn, M.D., University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D., V.A. Medical Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D., Ph.D, [https://archive.ics.uci.edu/ml/datasets/Heart + Disease](https://archive.ics.uci.edu/ml/datasets/Heart+Disease) (accessed 18.02.20).
- [23] <https://www.mayoclinic.org/diseases-conditions/chronic-kidney-disease/symptoms-causes/syc-20354521> (accessed 28.02.20).
- [24] D. Dua, C. Graff, UCI machine learning repository [mL]. School of Information and Computer Science, California, https://archive.ics.uci.edu/mL/datasets/Chronic_Kidney_Disease, 2019 (accessed 19.02.20).
- [25] <https://www.cancercenter.com/cancer-types/breast-cancer/> (accessed 20.02.20).
- [26] https://www.cancercenter.com/cancer-types/breast-cancer/stages?invsrc=paid_search_google_national_nonbrand_cancertype_breast&t_pur=prospecting&t_src=google-g&t_con=stages&t_re=national&t_bud=corporate&t_tar=non_targeted&t_aud=any&t_cam=1604505100&t_adg=63538112289&t_trm=%2Bbreast%20%2Bcancer%20%2Bstages&t_mtp=b&t_pos=&t_ctv=305585880147&t_d=-c&t_plc=kwd-296278002169&t_mkt=g-9028692&t_med=online&t_ch=paid_search&t_mdm=click&-kxconfid=s8ymtai82&dkstrackerid=43700037991404397&aws-earchepc=1&gclid=EAIAIqobChMI-riQo6vj5wIVhKDsCh0IHAupEAAAYAiAAEgI3g_D_BwE&gclidsrc=aw.ds (accessed 21.02.20).
- [27] M. Zwitter, M. Soklic, UCI machine learning repository (<http://archive.ics.uci.edu/mL>). University of California, School of Information and Computer Science, Irvine, CA, [https://archive.ics.uci.edu/mL/datasets/Breast + Cancer](https://archive.ics.uci.edu/mL/datasets/Breast+Cancer) (accessed 20.02.20).
- [28] <https://www.cancercenter.com/cancer-types/breast-cancer> (accessed 21.02.20).
- [29] <https://www.ibm.com/cloud/watson-studio/autoai> (accessed 29.02.20).
- [30] <https://www.ibm.com/cloud/machine-learning> (accessed 29.02.20).
- [31] <https://www.mathworks.com/products/statistics.html> (accessed 29.02.20).
- [32] <https://www.mathworks.com/help/stats/classificationlearner-app.html> (accessed 29.02.20).

Further reading

- <https://www.analyticssteps.com/blogs/introduction-linear-discriminant-analysis-supervised-learning> (accessed 28.02.20).
- <https://www.mathworks.com/help/stats/regression-learner-app.html> (accessed 29.02.20).
- https://www.researchgate.net/publication/254559798_Comparison_of_crisp_and_fuzzy_KNN_classification_algorithms (accessed 29.02.20).
- K.H. Padil, N. Bakhary, H. Hao, The use of a non-probabilistic artificial neural network to consider uncertainties in vibration-based-damage detection, *Mech. Syst. Signal. Process.* 83 (2017) 194–209. Available from: <https://doi.org/10.1016/j.ymssp.2016.06.007> ISSN 0888-3270. Available from: <http://www.sciencedirect.com/science/article/pii/S0888327016301893>.

Deep Learning Models Principles Applied to Biomedical Engineering

5.1 Deep learning based on artificial neural networks

“*Deep Learning (DL)*” is a subset of “*Machine Learning (ML)*,” and “*ML*” is a subset of “*Artificial Intelligence (AI)*” as shown in Fig. 1.3. There are two main reasons to differentiate the use of algorithms of “*DL*” to “*ML*”: “*Decision boundary*” and “*Feature engineering*,” where:

- “*Decision boundary*” in “*ML algorithms*” learns the mapping from input to output based on a set of weights in the architecture. For example, in “*ML classification problems*” the algorithm learns the “*decision boundary*” with a function to separate the classes. There is a “*linear decision boundary*” that resolves the determination of whether a given value point belongs to one class or the other based on linear functions, such as the “*Sigmoid function*” used in “*Logistic regression*” to separate the classes. Thus “*ML*” is not capable of learning complex types of relationships. The “*DL algorithms*” are designed to learn the “*nonlinear decision boundary*” in complex types of relationships.
- “*Feature engineering*” is the way of extracting features from data and transforming them into formats that are suitable for “*AI algorithms*.” The “*feature engineering*” goal is to turn data into information, and information into insight. It consists of two steps: “*feature extraction*” is the process of extracting all the required features necessary to resolve the specific problem; and “*feature selection*” is the process of selecting the important features for it. In an “*ML algorithm*” the feature extraction selection is made manually, and the feature selection is made automatically for the classifier, whereas in a “*DL algorithm*” both selections are made automatically. For example, extracting features manually from images is a tedious time-consuming task and requires a lot of knowledge from the programmer.

“*DL*” is inspired by the structure and function of “*neurons*” from the human brain, representing them as “*Artificial Neural Networks (ANN)*,” as illustrated in

Fig. 1.6A. They are designed to recognize patterns that are represented by numeric vectors that represent images, sounds, text, time series, processes, etc. The “*ANN*” is based on “*learning processes*” as a way of acquiring knowledge using parameters such as the “*synaptic weight*” and “*bias*” in the network that are adapted through a continuous simulation process by the environment in which the “*ANN*” is embedded, as explained in Fig. 1.6B. The most common “*ANN*” is the “*basic three-layered neural network*” defined as: “*input*,” “*hidden*,” and “*output*,” as shown in Fig. 5.1A, and its representation in Fig. 5.1B. The “*ANN*” calculates the values of the “*neurons in the output layer (y_k)*” based in the “*synaptic weights (w)*” and “*bias (b)*” in the “*neurons in the hidden layer ($W_{l,i}$)*” related to the values of the “*neurons in the input layer (x_i)*” compared with the expect “*targets values*,” as indicated in Eq. (5.1):

Artificial Neuron Network general equation

$$y_k = f \left(\sum_{i=1}^n (w_{l,i} + b) x_i \right) \quad (5.1)$$

where y_k is the number of response variables to predict; x_i is the number of predictor variables; $w_{l,i}$ is the weight of each connection in the hidden layer; b is the bias; m is the layer number; and f is the transfer function.

In this chapter, we discuss the basic “*artificial neural network*” and the “*deep neural network*,” where there are many multiple hidden layers, as indicated in Fig. 5.1C, that allow the computation of much more complex features of the input. This is based on each hidden layer computing a nonlinear transformation of the previous layer, representing a significantly greater power based on more complex functions, as represented in Fig. 5.1D. Therefore it is used a nonlinear activation function in each hidden layer.

The underlying principle of “*Deep Learning*” is that of the compositional nature of “*neural networks*” inspired by the biological elements that forms the “*human brain*,” such as a collection of “*nodes*” emulating “*brain neurons*”

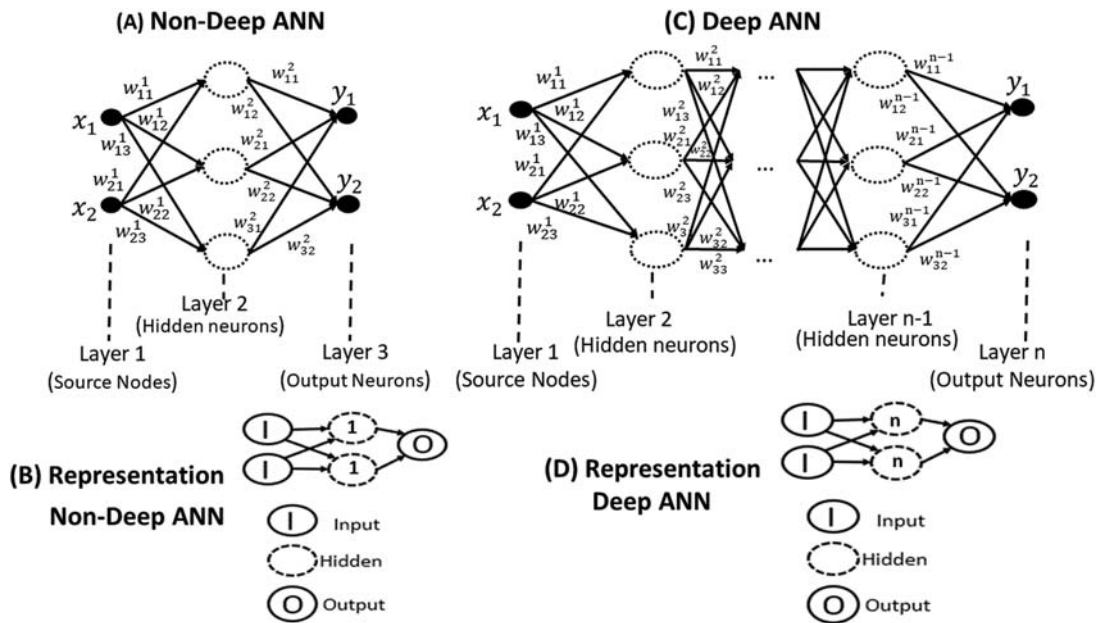


FIGURE 5.1 Artificial neural network (ANN): (A) non-deep ANN, (B) representation of non-deep ANN, (C) deep ANN, and (D) representation of deep ANN.

and their “neuron synapses connections as primary elements, that combine to form midlevel elements identified as “Artificial Neural Networks (ANNs),” which in turn are combined with different architectures to form more complex networks. These “ANN” are organized based on their architectural type, and the way their different components are connected to one another, defining the specific learning goal in different types*. They are classified as shown in figure 1.7*: “Feed Forward Neural Network,” “Backpropagation Neural Networks,” “Recurrent Neural Networks,” “Memory Augmented Neural Networks,” “Modular Neural Networks,” and “Evolutionary Neural Networks.” The first two “ANN types” are studied in depth in this chapter and the others are covered in the Chapter 6, “Deep Learning Models Evolution Applied to Biomedical Engineering.”

In this chapter the focus is on studying:

- “Feed Forward Neural Network” types: “Perceptron (P),” “Multilayer Perceptron (MLP),” “Radial Basis Network (RBF),” “Probabilistic Neural network (PNN),” “Extreme Learning Machine (ELM),” and others.
- “Backpropagation neural networks” types: “Auto Encoder (AE),” “Variational Auto Encoder (VAE),” “Denosing Auto Encoder (DAE),” “Sparse Auto Encoder (SAE),” “Deep Convolution Network (DCN)” or “ConvNet (CNN),” “Deconvolutional Network (DN),” “Deep Convolutional Inverse Graphics Network (DCIGN),” “Generative Adversarial Network (GAN),” “Deep Residual Network (DRN) or Deep ResNet,” and others.

- “Shallow Neural Networks,” as a way to introduce and explain the “deep learning models principles applied to biomedical engineering.”
- “Transfer learning from pretrained deep learning networks.”

All examples are based on practical research on biomedical engineering using existing AI tools from: “MATLAB® and “IBM Watson Studio.”

Note*: There is no standard set of rules to classify the different types of “ANN,” nevertheless with the objective of organizing the study and application of “ANNs” for the “AI analysis” and obtention of “AI models” in many different research areas of “Biomedical Engineering,” in this book the “ANNs” are organized based on their architectural type and the way their different components are connected to one another, defining the specific learning goal. They are separated into six different types, as shown in Fig. 1.7.

5.2 Feed forward neural networks types

“Feed Forward Neural Network” implies a signal that can only be fed forward, meaning the absence of recurrent or feedback connections. In other words, the data path in the network is only forward facing, no backward feed connections between neurons are present. Some frequently used examples of “Feed Forward Neural Network” are shown in Fig. 1.8, these are: “Perceptron (P),” “Multilayer

“Perceptron (MLP),” “Radial Basis Network (RBF),” “Probabilistic Neural network (PNN),” “Extreme Learning Machine (ELM),” and others.

5.2.1 Perceptron (P) or single-layer perceptron network

“P” is the simplest of “the ANNs,” that consists of a “single layer of output nodes” that represent the “neurons,” where they are fed directly from the “input layer nodes” using a series of “weights,” which show the strength of each particular node, as shown in Fig. 5.2A. The sum of the products of the weights and the inputs is calculated in each node, based on the “activation function” which is explained in the Section 5.2.1.1 In the “activation function” is defined a “threshold value” that typically is zero, if the node value is above the “threshold value” the neurons fire and take the activated values, which are typically “+1,” otherwise they take the deactivated values, which is typically “−1 or 0 in binary,” as shown in Fig. 5.2B. These actions are used to represent the “synapse” between two human brain neurons, as indicated in Fig. 1.6. “Perceptron” is frequently used to classify the data into “n parts,” see the example in Fig. 5.2D. Therefore it is also known as a “Linear Binary Classifier.”

“Perceptron” is a basic concept that can be applied in many application, such as hybrid collision detection perceptron of the robot in the fusion application [1], progressive operational perceptrons with memory [2], deep belief network and linear perceptron-based cognitive computing for collaborative robots [3], and many more.

Example 5.1: Example of “Perceptron” and threshold activation function.

We need to classify patients from a hospital that have diseases based on age in three ranges: “Infant diseases when ≤ 2 years old,” “Middle age diseases when $45 < \text{age} < 65$,” and “Elderly diseases age when $\text{age} \geq 65$.”

The solution proposed is based on the “Perceptron” network, as shown at the bottom of Fig. 5.2D, in order to obtain a classification based on “clusters”, as shown at the top of Fig. 5.2D.

5.2.1.1 ANN activation functions

The “activation functions” are used to map the input between the required values, like (0, 1) or (−1, 1). Then, they can be basically divided into two types of functions:

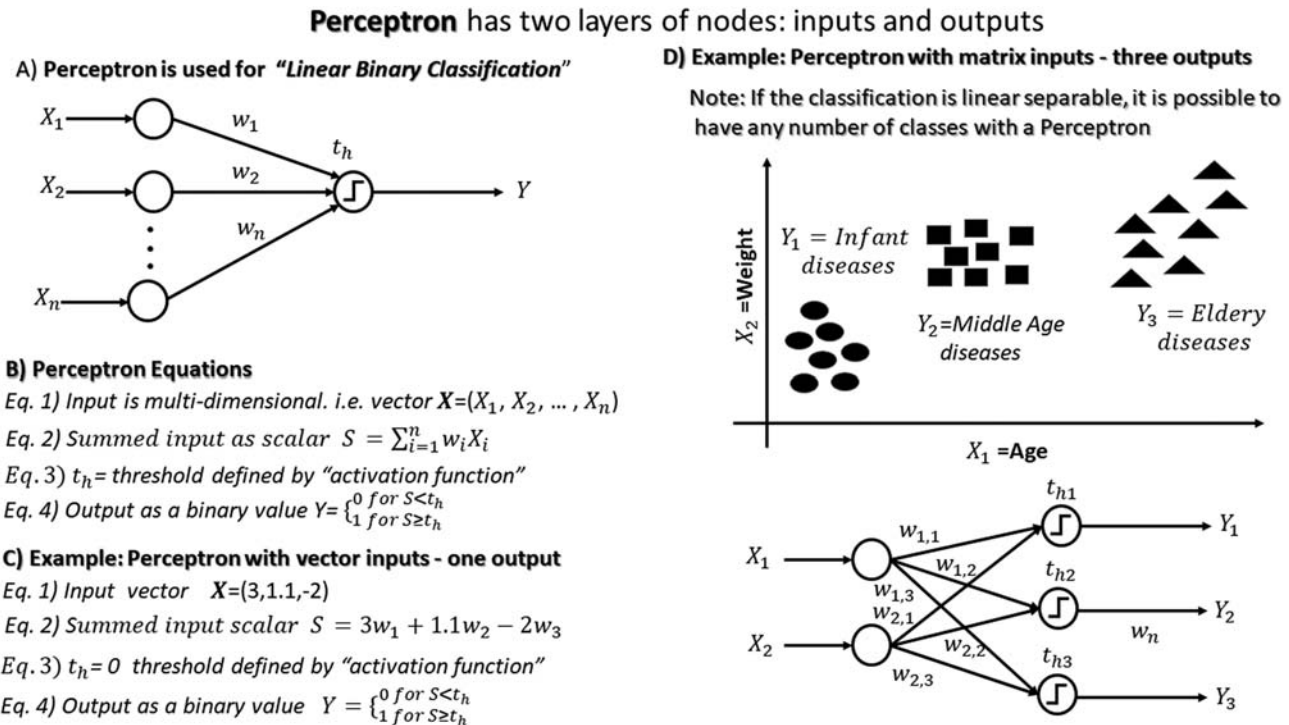


FIGURE 5.2 Perceptron: (A) linear binary classification, (B) perceptron equations, (C) example of perceptron with a vector input and one output, and (D) example of perceptron with matrix inputs and three outputs.

Activation Function									
	Identity	Binary Step	Logistic	TanH	ArcTan	ReLU	PreLU	ELU	SoftPlus
P L O T									
E Q U A T I O N	$f(x)$ = x	$f(x)$ = $\begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f(x)$ = $\frac{1}{1 + e^{-x}}$	$f(x)$ = $\frac{2}{1 + e^{-2x}} - 1$	$f(x)$ = $\tanh(x)$ = $\tan^{-1}(x)$	$f(x)$ = $\begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f(x)$ = $\begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f(x)$ = $\begin{cases} \alpha(e^{-x}-1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f(x)$ = $\log_e(1 + e^x)$
D E R I V A T E	$f'(x)$ = 1	$f'(x)$ = $\begin{cases} 0 & \text{for } x \neq 0 \\ \delta & \text{for } x = 0 \end{cases}$	$f'(x)$ = $f(x)(1-f(x))$	$f'(x)$ = $1-f(x)^2$	$f'(x)$ = $\frac{1}{x^2 + 1}$	$f'(x)$ = $\begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x)$ = $\begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x)$ = $\begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x)$ = $\frac{1}{1 + e^{-x}}$

FIGURE 5.3 Activation function used on Artificial Neural Networks.

“linear activation” and “nonlinear activation.” Some of the most frequent “activation functions” used in “ANNs” for linear activation are “identity,” and for nonlinear activation they are “Binary step,” “Logistic,” “TanH,” “ArcTan,” “Rectified Linear Unit known (RELU),” “Parametric Rectified Linear Unit (PreLU),” “ELU,” “Soft plus,” and others such as “Sigmoid” (describes a general class of curves that are “S-shaped,” such as the “Logistic” and “Tanh,” as shown in Fig. 5.3) and “Softmax,” which is frequently used in “AI classifiers” and will be explained later in this chapter. The “activation functions” are summarized in Fig. 5.3 showing their function plot, equation, and their derivative function with their respective ranges

There are other activation functions that can be used as: approximation rates for neural networks with general activation functions [4], weighted sigmoid gate unit for an activation function of deep neural network [5], global exponential stability of delayed complex-valued neural networks with discontinuous activation functions [6], and many others that can improve networks for many different application in biomedical engineering.

5.2.2 Multilayer perceptron

“Multilayer perceptron (MLP), also called Feed forward Neural Network (FFNN),” is shown in Fig. 5.4A. “FFNN”

are arranged in layers, with the first layer connected to “inputs” and the last layer producing “outputs.” The middle layers have no connection with the external world, and hence are called “hidden layers.” It is possible to increase the number of hidden layers as needed to make the model more complex according to the problem being resolved. When we have more than three hidden layers they are known as “Deep Feed forward Network (DFFN),” as shown in Fig. 5.1B. The goal of an “MLP” is to approximate the function $y = f^*(x)$ that maps the input “x” to a class “y” to obtain the classification needed. For example, the “MLP” with three layers have three functions represented as: $f(x) = f(3)f(2)f(1)$. Each layer is represented as: $y = f(w_i X_i + b)$, where “f” is the activation function, “w_i” are the weights in the layer, “X_i” are the input vector, and “b” is the bias vector. The layers of an “MLP” consist of several fully connected layers because each unit in a layer is connected to all the units in the previous layer. In a “fully connected layer,” the parameters of each unit are independent of the rest of the units in the layer, that means each unit possesses a unique set of weights. In “supervised classification” the output includes a “class score.” Then, the network performance as a “classifier” is obtained by a “loss function,” when the loss value is high the predicted class is not trustable, and when it is low the predicted class is a true class. A frequent strategy consists of initializing the “weights” to random values and refining them iteratively to get a lower loss. This is the reason that the training of an “MLP” or “DFFN” is made in three steps: “forward propagation,” “loss calculation,” and “backward

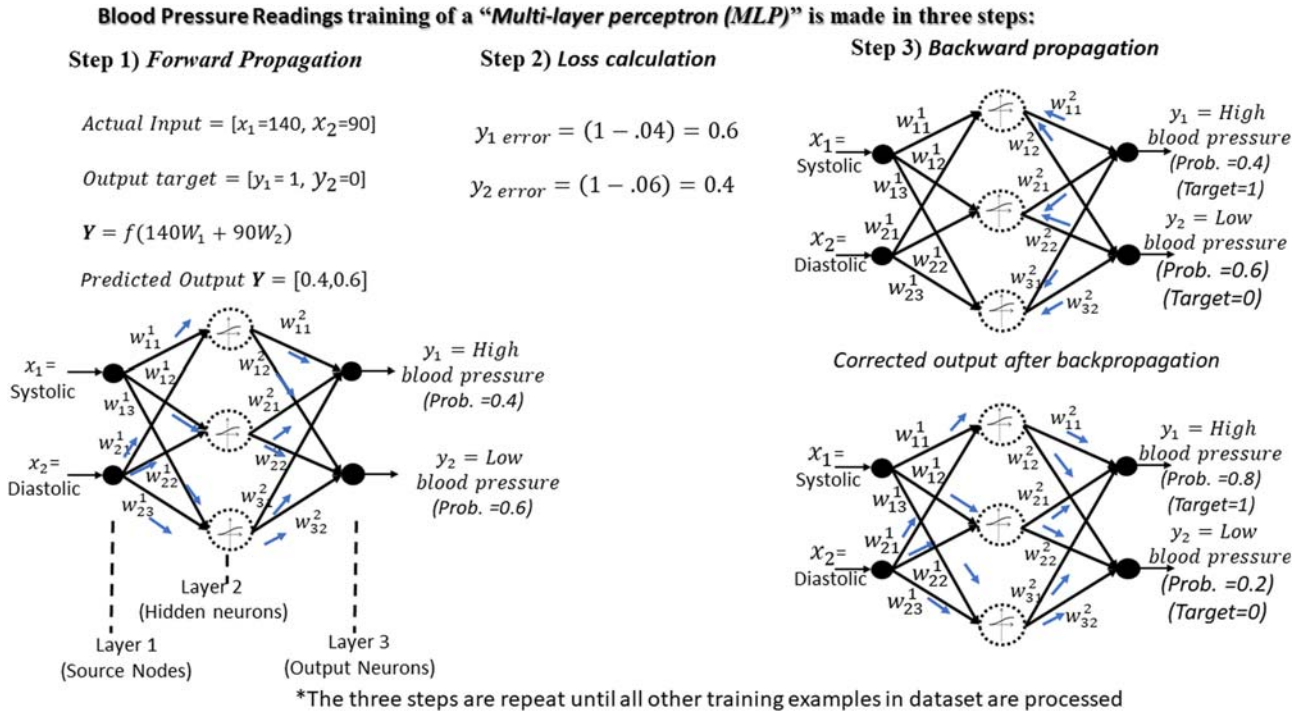


FIGURE 5.4 Example of “multilayer perceptron” for “blood pressure reading.”

propagation*” as shown in the example for “blood pressure reading” in Fig. 5.4.

1. “Forward propagation”: In this step the training is made by passing the input to the model, multiplying “weights” assigned randomly and adding “bias” at every layer to find the output value of the model, as shown in Eq. (5.1).
2. “Loss calculation”: After step 1 the output values for the model are obtained, they are called “Predicted output” and are compared with the “target value” or “expected output.” This difference is used to calculate the “loss value” by the “loss function.”
3. “Backward propagation” is when the errors obtained are sent back through the network calculating the “Gradients.” Then an optimization method, such as “Gradient descent” calculates and adjusts all weight in the backward process applying “gradient flow” in that direction where the goal is to reduce the error at the output layer.

Note*: With the goal to explain “MLP and the direct relation with the Perceptron as a “FFN,” “MLP is studied in the category of an “FFN,” nevertheless “MLP” has a training step with a “backpropagation algorithm” and it should be placed in “Backpropagation Neural Network” category.

These steps are repeated iteratively moving the weight refinement in the direction defined by lower loss function

values. The activation functions frequently used are “Logistic (Sigmoid),” “Relu,” and “TangH,” as shown in Fig. 5.3.

Example 5.2: Example for MLP of analyzing “blood pressure reading.”

The “blood pressure reading” example shown in Fig. 5.4 has a process in three steps applying a “MLP” network to identify when a person has “high blood pressure” or “low blow pressure.” It is based on “systolic” blood pressure, which indicates how much pressure your blood is exerting against artery walls when the heart beats, and “diastolic” blood pressure, which indicates how much pressure your blood is exerting against artery walls while the heart is resting between beats. A normal blood pressure reading is 120/80 mm Hg. The three steps for a “MLP” are:

- Step 1) “Forward propagation” as defined in the equations in indicated in this step at Fig. 5.4
- Step 2) “Loss calculation function” as calculated in equations shown in indicated in this step at Fig. 5.4
- Step 3) “Backpropagation algorithm” as indicated in two substeps in Fig. 5.4C as they correct the values backwards and forward.

The three steps are repeat until all other training examples in dataset are processed.

The “MLPs” can be used in many problems in Biomedical Engineering such as using multilayer perceptron with Laplacian edge detector for bladder cancer diagnosis [7], hybrid particle swarm optimization-genetic algorithm trained multilayer perceptron for classification of human glioma from molecular brain neoplasia data [8], classification of cervical cancer using hybrid multilayered perceptron network trained by genetic algorithm [9], and many others that combine “MLPs with others AI models.”

5.2.3 Radial basis function network

“MLP” is applied in supervised learning when data is not linearly separable. Otherwise complex nonlinear classifiers can be built by combining them into a network, such as the “Radial Basis Network (RBF),” that can be used in nonlinear classifications and other applications as “RBF function approximation.” It performs classification by measuring the input’s similarity to examples from the training set.

Each “RBF neuron” stores a “prototype,” which is just one of the examples from the training set. Where the goal is to classify a new input, each neuron computes the “Euclidean distance” between the input and its prototype. If the input more closely resembles the “class A” prototypes than the “class B” prototypes, it is classified as

“class A,” and so on. “RBF” has many applications, such as function approximation, time series prediction, classification, and system control. A typical “RBF network” consists of an “input vector (x_i),” a layer of “RBF neurons (φ),” and an “output layer (y_i),” as shown in Fig. 5.5A.

- “Input vector” is a n -dimensional vector that needs to be classified.
- “RBF neurons” is a layer where each neuron stores just one of the “prototype vectors” from the training set and each is compared to the actual “input vector” and delivers a value from an “RBF function activation” that represents the similarity between them with values from “0 for no similarity” to “1 for complete similarity.” As the distance between the input and prototype grows, the response falls off exponentially, forming a “bell curve,” see Fig. 5.5B, that is calculated using the equations shown in Fig. 5.5C. β controls the width of the bell curve and σ is calculated by the “Euclidian distance” as $\|x_i - \mu\|^2$.
- “Output layer” has one node for each class of data. Each output node computes the score for a different class; as a consequence every output node has its own set of weights, as shown in Fig. 5.5D.

Note: The output node will typically give a positive weight to the RBF neurons that belong to its category, and a negative weight to the others.

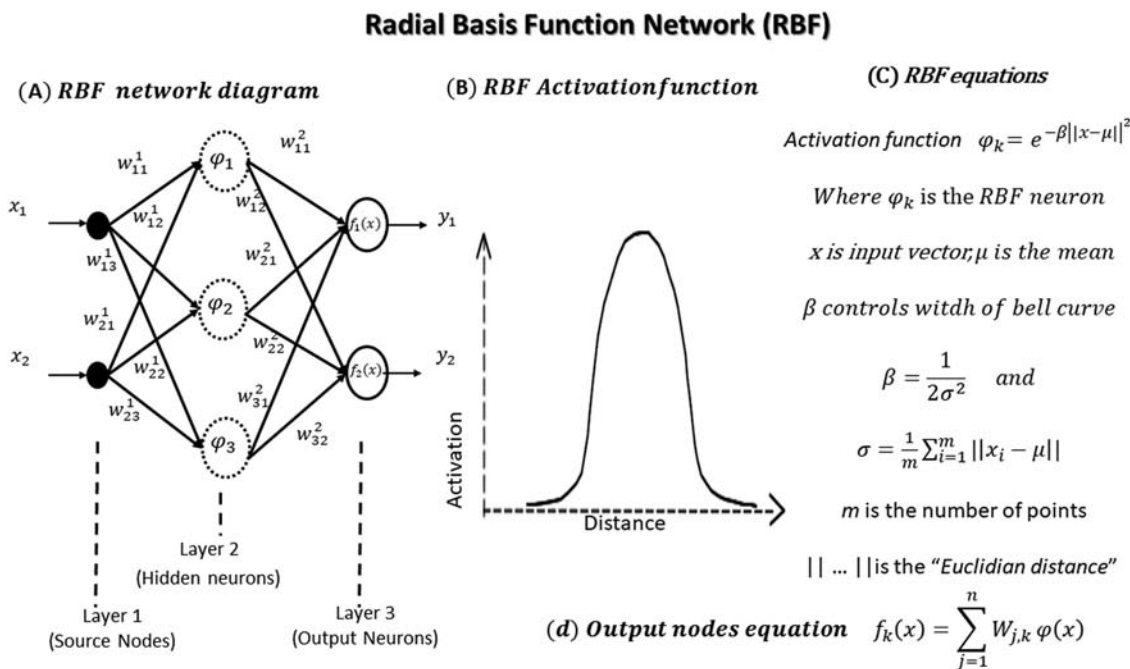


FIGURE 5.5 “Radial Basis Function (RBF)”: (A) typical network diagram, (B) typical “RBF” activation function, (C) “RBF” equations, and (D) “RBF” nodes equation.

Example 5.3: Example of “RBF” to analyze “heart rhythm.”

Problem to resolve in this example

A patient has taken 30 consecutive readings of his “heart rhythm.” The readings in the dataset are normalized from “0 to 1.” Obtain an “RBF approximation” model to predict if he has a “normal heart rate” or not. The “normal heart rate” is usually stated between “60 and 100 beats per minute.” Slower than 60 is known as “bradycardia” and faster than 100 is identified as “tachycardia.”

Proposed solution to analyze “heart rhythm” using “RBF”

An “RBF model” is proposed, with a MATLAB script using the “Deep Learning Toolbox.” Open the MATLAB program and go to the book data companion directory: “... \Exercises_book_ABME\CH5\MATLAB_RBF” and open “RadialBasisFunctionFunction.m.”

The MATLAB script “RadialBasisFunctionFunction.m,” as shown in Fig. 5.6A, is divided into five steps. These are

- Step 1) Load and plot dataset. Here the 30 consecutive normalized between [0...1] “heart rhythm” readings are specified in the script.
- Step 2) Plot the “RBF Activation Function,” a plot of the “RBF approximation function” is obtained showing the “bell curve.”

- Step 3) Define “RBF network” and view configuration. In this step the precision of the “RBF” is defined for the “sum-squared error goal” and the spread constant. Then the “RBF” diagram is generated showing one input vector, one hidden layer with 26 “RBF nodes,” and a layer output with one node.
- Step 4) Plot “RBF results and expected values.” A plot showing the target values with “+” symbol and the approximation results as a continuous type line.
- Step 5) Predict values using “RBF model.” Because of the small amount of reading in this example, only predictions for close next values are valid. The result seems to indicate the “heart rhythm” is becoming stable in approximately 0.5 indicating a stabilization in the “normal heart rhythm.”

“RBF” has been used in many biomedical engineering applications, such as automated diabetic retinopathy detection using radial basis function [10], image reconstruction for electrical impedance tomography: experimental comparison of radial basis neural network and Gauss–Newton method [11], some extensions of radial basis functions and their applications in artificial intelligence [12], and many other applications.

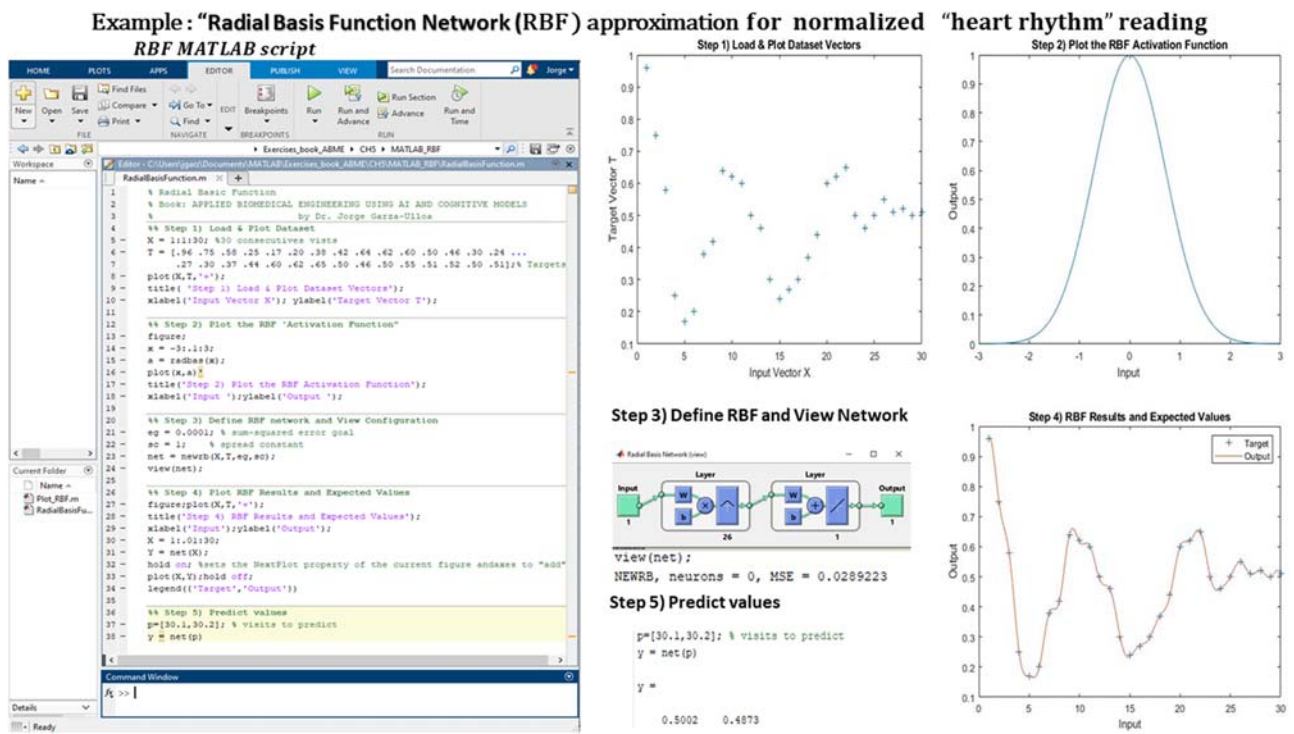


FIGURE 5.6 Example of “RBF approximation” for normalized “heart rhythm” readings.

5.2.4 Probabilistic neural network

A “probabilistic neural network (PNN)” is a four-layered feed forward neural network used for classification, the net architecture is shown in Fig. 5.7A. The “PNN layers,” are “input,” “one hidden,” “one linear (pattern/summation),” and “output” [13]:

- “Input layer” has the source’s nodes.
- “One hidden radial basic layer” computes the distances from the input vector to the training vector and produces a vector whose elements indicate how close the input is to a class using an “RBF function activation.”
- “One summation/competitive layer” that sums all the contributions for each class producing a “net output” as a “vector of probabilities” using “probability distribution function (PDF)” and a “competitive transfer function” pick the maximum of these probabilities producing a class indicated by “1” or “0.”
- “Output layer” contains the vector with the assigned class as “1” or “0” for the result for the inputs.

Example 5.4: Example of “probabilistic neural network (PNN)” to classify the “types FLU: A, B, and C.”

“Flu” or “influenza” is a contagious respiratory function caused by a variety of flu viruses. The symptoms involve “muscle pain,” “headache and fever.” Assume, that there are three types of flu viruses that affect humans: “A,” “B,” and “C” [14,15]:

- “Type A” causes seasonal flu epidemics practically every year. It can infect humans and animals.

“Influenza A” is the only type that can cause a “pandemic,” which is a global spread of disease. Bird flu and swine flu pandemics both resulted from “influenza A viruses.” An “influenza A virus” has two surface proteins: “hemagglutinin and neuraminidase.”

- “Type B” can also cause seasonal epidemics that typically only affect humans. There are two lineages of influenza B: “Victoria and Yamagata.” Influenza B viruses mutate more slowly than influenza A viruses.
- “Type C” causes less severe flu symptoms. It causes mild illnesses, but they do not appear to cause epidemics.

Assuming “only for this hypothetical example (in reality it is far more complicated)” that:

- Type “A” = “class 1” had “strong headache and higher fever,” and “mild muscle aches and ache.”
- Type “B” = “class 2” had “strong muscle pain ,” and “mild headache and fever.”
- Type “C” = “class 3” had “lesser headache and fever” and “lower muscle pain”

Problem to resolve in this example

Obtain an “AI model using PNN” for classification according to the symptom’s severities.

Proposed solution for influenza types hypothetical classification using “PNN”

A “Probabilistic Neural network (PNN) model” is proposed, with a with a MATLAB script using the “Deep Learning Toolbox.” Open the MATLAB program and go to the book data companion

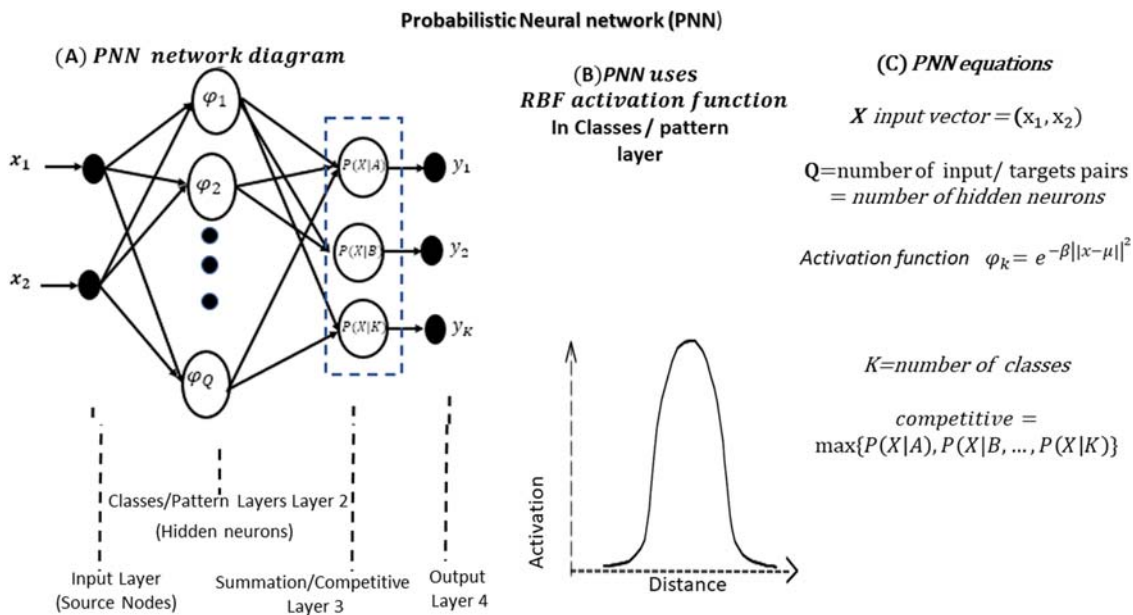


FIGURE 5.7 Probabilistic neural network (PNN): (A) PNN network diagram, (B) “PNN” uses RBF activation function in classes/pattern layer, and (C) PNN equations.

Example: “Probabilistic Neural Network (PNN) for hypothetical input data for “Types FLU: A, B, and C”

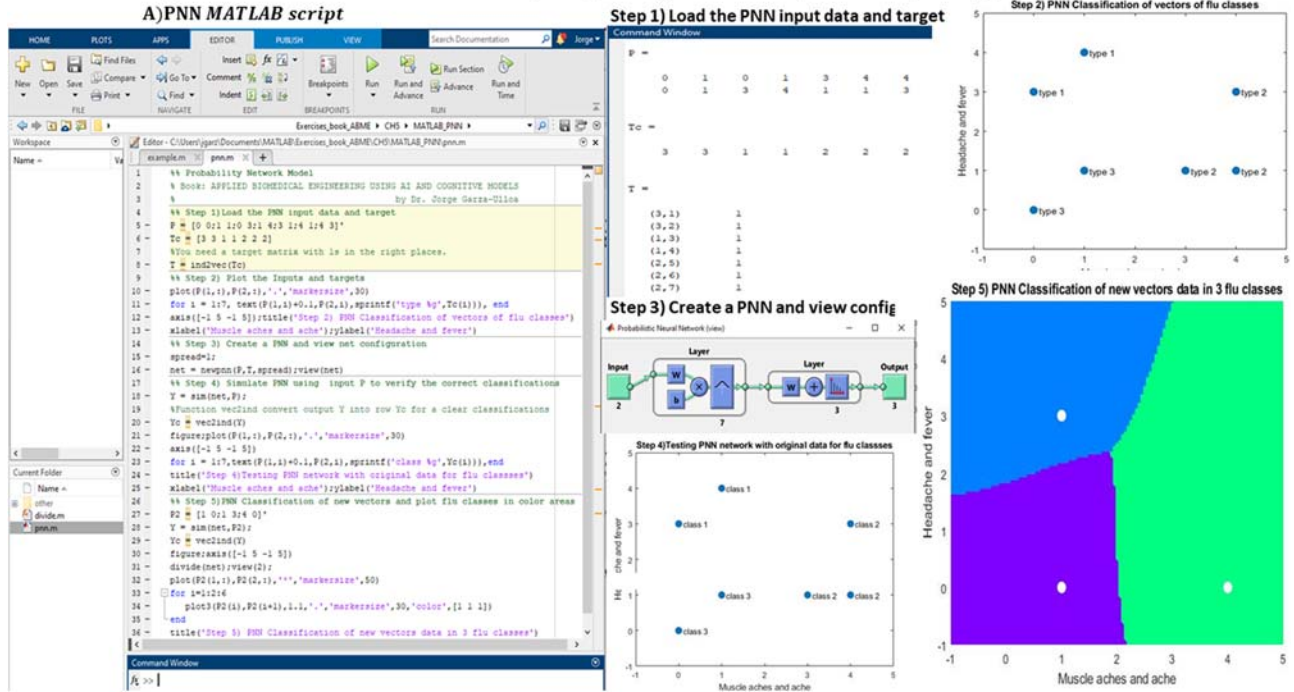


FIGURE 5.8 Example “PNN for hypothetical input of “types of flu: A, B, and C”: a) PNN MATLAB script, and steps 1) to 5) to plot the results.

directory: “...Exercises_book_ABME\CH5\MATLAB_PNN” and open the script “pnn.m.”

The MATLAB script “pnn.m” is divide in five steps as shown in Fig. 5.8A).

- Step 1) Load the “PNN input and target data,” seven vector pairs with their expected classes.
- Step 2) Plot the inputs and targets, a plot for “PNN classification of vectors of flu classes” is generated.
- Step 3) Create a “PNN and view its net configuration.”
- Step 4) “Simulate PNN using input P to verify the correct classifications,” a plot using the net “PNN model for classification” of vectors of flu classes is generated.
- Step 5) “PNN classification of new vectors” and plot classes in color areas, the three areas for each type of flu are shown with the results for the new vectors.

“PNN” has been applied in research in many different fields of biomedical engineering, such as training the probabilistic neural network to solve classification problems [16], small lung nodules detection based on local variance analysis and probabilistic neural network [17], child emotion recognition using probabilistic neural network with effective features [18], and many others.

5.2.5 Extreme Learning Machine

“Extreme Learning Machine (ELM)” is a method that is essentially a single feed forward neural network; its structure consists of a single layer of hidden nodes, where the weights between inputs and hidden nodes are randomly assigned. This means that it does not need a learning process to calculate the parameters of the models, and remains constant during training and predicting phases. On the contrary, the weights that connect hidden nodes to outputs can be trained extremely fast [19]. The greatest advantage of “ELMs” is that they are cheap computationally for implementing online models [20]. ELM is used for pattern classification and function approximation. In summary, “ELM” is the simplest kind of neural network. It consists of three layers: an input layer, a hidden layer, and an output layer (as shown in Fig. 5.1A, as a “typical non-deep ANN”). An “ELM” is shown with more detail in Fig. 5.9A. The training of “ELM” is quite simple; it only needs to update the “output weights/beta” to be optimized as constraints using the equations at Fig. 5.9C, while the input weights “W and biases” of the hidden layer are randomly generated. “ELM theory shows that hidden neurons are important but need not be tuned in many applications,” for example, feature learning, clustering, regression, and classification. In theory, such neurons can be almost any nonlinear piecewise continuous neurons including hundreds of types of biological neurons of

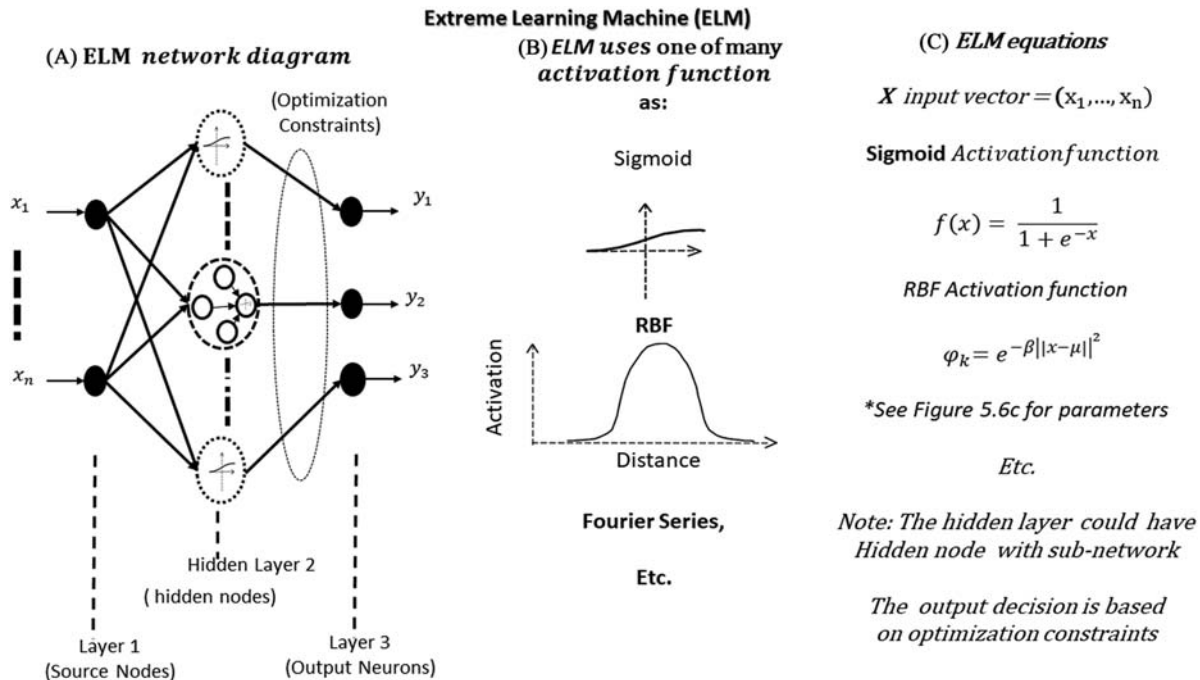


FIGURE 5.9 Extreme Learning Machine (ELM): (A) ELM network diagram, (B) ELM uses one of many activation functions available, and (C) ELM equations.

which the exact math modeling may be unknown to human being [21].

Example 5.5: Example of “Extreme Learning Machine (ELM)” for diabetes values prediction.

Problem to resolve in this example

There is a need to detect diabetes in Indian women based on parameters such as “age,” “diabetes pedigree function,” “body mass index,” “serum insulin,” “triceps skin fold thickness,” “diastolic blood pressure,” “plasma glucose concentration,” and “number of times pregnant.”

Objective of this example

Obtain an “ML prediction AI model using ELM ANN for diabetes dataset*” used in the Research 4.5, in MATLAB with Statistics and Machine Learning Toolbox installed.

Note *: In this example the dataset for “diabetes.csv” based on information at the “National Institute of Diabetes and Digestive and Kidney Diseases with title: Pima Indians Diabetes Database” (<https://www.niddk.nih.gov/>) is used to obtain the an “AI LM model.” This dataset has: “768 records,” with “eight fields (attributes)” described in Table 5.1.

For this example, all fields are normalized between 0 and 1, except for the “class” field, then all fields were exported as a “text tab delimited” under the name “diabetes_train.txt.”

Note: The dataset can be found in the companion set exercise for this book at: “... \Exercises_book_ABME \CH5\MATLAB_ELM\diabetes_train.txt.”

Proposed solution for predicting diabetes in Indian women using “Extreme Learning Machine (ELM)”

An “ELM to predict diabetes” is proposed with a MATLAB script. Open the MATLAB program and go to the book data companion directory: “... \Exercises_book_ABME\CH5\MATLAB_ELM” and open the script “ELM_main.m.” This script is divided into four steps as shown in Fig. 5.10A). These are:

- Step 1) “Load input data with targets,” the normalized dataset provided as “diabetes_train.txt” is loaded in variable “train_data.”
- Step 2) “Specify model parameters and obtain ELM model.” The variables for the type of ELM is set as classifier specifying “elm_type = 1,” if you wish a regression define “elm_type = 0,” the number of neurons of hidden neurons are defined as: “NumberofHiddenNeurons = 1000,” and the activation function is specified as: “ActivationFunction = sig,” where the activation functions available are: “sig = sigmoid,” “sin = sine,” “hardlim.” “Elm_training function” is called, the “training accuracy” and the “training time” are display, and the “ELM net model” is available in the variable “elm_model.mat.”
- Step 3) “Load the test data with targets from the text file diabetes_test.txt” in the variable “test_data.”
- Step 4) “Execute ELM predict and obtain a prediction with the user function elm_predict.m.” The results are: “TestingTime = ~0,” “TestingAccurray = 0.75.” All results are saved in the variable “elm_output.m.”

TABLE 5.1 Diabetes dataset for diabetes values prediction.

Field	Description * 768 instances of patients. all Females >21 and <81 years old
class	Boolean Class variable: ['0 = tested_negative', 1 = tested_positive']
preg	Number of times pregnant (integers)
plas	Plasma glucose concentration 2 h oral glucose tolerance test
pres	Diastolic blood pressure in mm Hg (real number)
skin	Triceps skin fold thickness in mm (real number)
Insu	2-H serum insulin in mu U/mL (real number)
mass	Body mass index based on weight in kg/(height in m) ² (real number)
pedi	Diabetes pedigree function (real number). It is a function which scores the likelihood of diabetes based on family history
age	Age in years (integers)

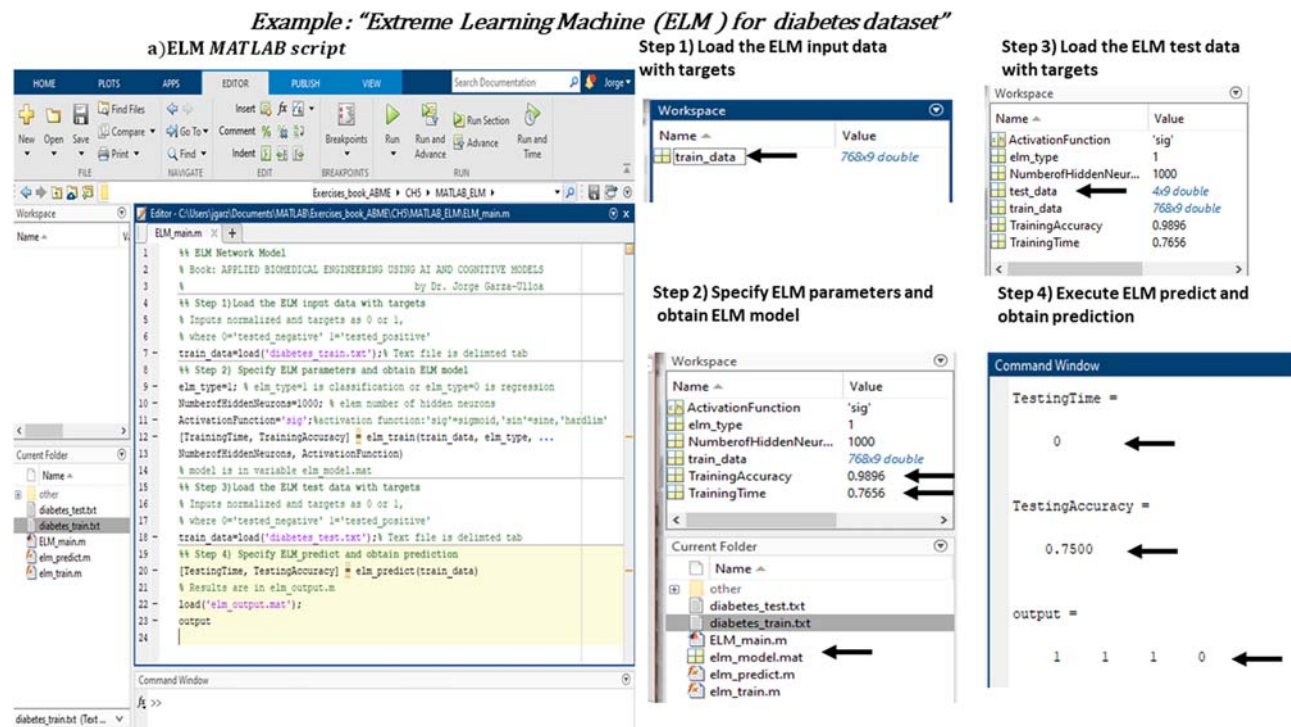


FIGURE 5.10 “ELM example for diabetes dataset”: 1) Load input data with targets, 2) Specify model parameters and obtain ELM model, 3) Load the test data with targets, and 4) Execute ELM predict and obtain prediction.

Note: In this case the output vector result obtained is: [1 1 1 0] instead of the expected result of [0 1 1 0] because the dataset is small with “testing accuracy = 0.75.”

In summary, “ELM” is a simple learning algorithm for “Single-Layer Feed Forward Neural Network (SLFN).” In theory, the “ELM algorithm” tends to provide good
(Continued)

(Continued)

performance at extremely fast learning speed. Unlike “traditional feed forward network learning algorithms” like “back-propagation (BP) algorithms,” the “ELM” does not use a “gradient-based technique.” With this method, all the parameters are tuned once, and this algorithm does NOT need iterative training, which is the reason for its fast answer [22].

(Continued)

(Continued)

It has been applied in many problems in biomedical engineering, such as a novel randomized machine learning approach: reservoir computing extreme learning machine [23], deep and wide feature based extreme learning machine for image classification [24], classifier for face recognition based on deep convolutional–optimized kernel extreme learning machine [25], discrimination of β -thalassemia and iron deficiency anemia through extreme learning machine and regularized extreme learning machine based decision support system [26], and many others.

5.3 Shallow neural network

“Shallow neural networks” are special neural networks that consist of only “one or two hidden layers.” Understanding a “shallow neural network” gives us an insight into what exactly is going on inside a “deep neural network” that has “three” or more hidden layers.”

“Shallow neural networks” can be used in many ways for different purpose, such as:

- “Feed Forward Neural Network Fitting,” as explained at the “Section 5.3.1, Research 5.1 MATLAB Deep Learning Toolbox using a Feed Forward neural network fitting for human body fat.”
- “SOM Neural Network” as explained at the “Section 5.3.2, Research 5.2 MATLAB Deep Learning Toolbox using a neural network for clustering based on self-organizing MAP to analyze surface electromyography signals.”
- “Neural Network-Nonlinear Autoregressive” as explained at the “Section 5.3.3, Research 5.3 MATLAB Deep Learning Toolbox using a Neural Network for dynamic time series based on a nonlinear autoregressive to analyze vertical ground reaction forces signals.”
- And many other applications.

In summary, “Shallow Neural Networks” is a term used to describe “ANN” that usually have only one or two hidden layers as opposed to “DL NN” which have several hidden layers, often of various layer types, that with the right architectures achieve better results than the “shallow neural network.” The main difference is that the “deep models” can extract/build better features than “shallow models,” and to achieve this they are using the intermediate hidden layers. “Shallow neural networks” are used when the dataset’s have smaller labeled training data, for example, 50k docs, as “Shallow CNNs” work better than “Deep CNNs” [27].

5.3.1 Research 5.1 Feed Forward Neural Network to Analyze “Human Body Fat”

5.3.1.1 Case for research

Obtain an AI mode using a Feed Forward Neural Network (FFNN) to calculate the “Human body fat percentage” based on parameters such as “age,” “weight,” and some physical dimension of the human body, and assign the result to specific human body fat ranges.

5.3.1.2 General objective

Obtain a “Fitting Neural Network” AI model for “FFNN” as a MATLAB function to predict the “human body fat percentage” based on 13 input variables to obtain the “body fat percentage” from a US males dataset and classify them in the ranges of: “below normal range,” “normal range,” “overfat,” or “obese.”

5.3.1.3 Specific objectives

- Load the “Body_Fat_Estimation_Males_USA.csv” dataset for input of the “ANN,” and its “Target.csv” file as the data to be used to obtain the AI model.
- Define validation and testing groups dividing the samples randomly as: “70% for training,” “15% for validation,” and “15% for testing.”
- Select the optimal number of “Hidden layers” for the best “fitting FFNN.”
- Select the best algorithm for the “fitting FFNN.”
- Learn the best parameters criteria to detect the best performance model.
- Deploy the MATLAB “Fitting FFNN” model.
- Test the MATLAB “Fitting FFNN” model as a function matrix based on 13 input variables to obtain the “body fat percentage.”

5.3.1.4 Background for “Human Body Fat”

The most frequently used and accepted measure of “human body fat” is the “Body Mass Index (BMI),” as the relation between “height (in pounds)” and “weight (in inches)” defined by the “American Heart Association.” Their “BMI” defined ranges are shown in Table 5.2.

TABLE 5.2 Categories based on “Body Mass Index.”

Category	Body mass index
Underweight	< 18.5
Normal or healthy weight	18.5 and \leq 24.9
Overweight	25 and <30
Obese	\geq 30

However, “*BMI*” does not distinguish between lean muscle and fat mass, and “*BMI*” does not account for many factors, such as “*gender*,” “*age*,” “*ethnicity*,” and other useful variables, so it may not be an equally valid test for all populations. “*BMI*” does not produce an accurate description of human health and clinicians have proposed other methods based in the “*body fat percentage*” such as “*Skinfold measurements*,” “*Circumference measurements*,” “*Body fat scales*,” “*Dual-energy X-ray absorptiometry (DEXA)*,” “*Body Fat Percentage (BFP)*,” “*Hydrodensitometry*,” “*Air displacement plethysmography (ADP)*,” “*3D body scanners*,” and others:

- “*Skinfold measurements*” measure the thickness of skinfold in different areas of the body.
- “*Circumference measurements*” measure the circumference of different parts of the human body using a tape measure.
- “*Body fat scales*” measure the “*bioelectrical impedance analysis (BIA)*,” applying a very weak electrical current through the body to measure the body resistance; higher values indicate a higher body fat mass.
- “*Dual-energy X-ray absorptiometry (DEXA)*” uses “*X-rays*” to precisely measure the body fat, lean muscle, and mineral composition in different parts of the human body.
- “*Body fat percentage (BFP)*” measures the proportion of body fat composition. A healthy body composition must have a high proportion of fat-free mass due to muscles, bones, and human organs, and an acceptable low level of body fat.

- “*Hydrodensitometry*” or “*underwater weighing*” uses a person’s body mass and volume to calculate their body density measuring the volume of water that they displace.
- “*Air displacement plethysmography (ADP)*” uses air pressure sensors to measure the amount of air displaced to indicate body volume.
- “*3D body scanners*” uses lasers to create a “*3D image*” of the body to calculate the body volume.

Each method has its advantages and disadvantages, but in general, the “*body fat ranges*” obtained for any method are categorized in ranges of “*body fat percentage*.” For example, these ranges are shown for the United States in [Table 5.3](#).

5.3.1.5 Dataset

The dataset “*Body_Fat_Estimation.csv*” is based on the information of “*Circumference measurements*” from 252 males in the United States, with the fields (attributes) as indicated in [Table 5.4](#) and their targets in [Table 5.5 \[28\]](#).

5.3.1.6 Procedure

The steps to obtain an AI model “*using MATLAB Deep Learning Toolbox for NNF fitting apps of a body fat estimation dataset*” are summarized in [Table of slides 5.1](#) and each step of the example is visually explained using screen sequences with instructions in easy to follow screen figures.

TABLE 5.3 Categories based on “*Body Fat Ranges for United States*” as “*Body Fat Percentage*.”

Category	Women	Men
Below normal range	< 22%	< 15%
Normal range	20%–25%	15%–19%
Overfat	> 25% and ≤ 30	> 19% and ≤ 25%
Obese	> 30%	> 25%

TABLE 5.4 Dataset “Body_Fat_Estimation_Males_USA.csv” fields and descriptions.

Field	Description* 252 instances all males. Number of fields = 13
Age	Age in years
Weight	Weight in pounds
Height	Height in inches
Neck_circum	Neck circumference in inches
Chest_circum	Chest circumference in inches
Abdomen_circum	Abdomen circumference in inches
Hip_circum	Hip circumference in inches
Thigh_circum	Thigh circumference in inches
Knee_circum	Knee circumference in inches
Ankle_circum	Ankle circumference in inches
Biceps_ext_circum	Biceps (extended) circumference in inches
Forearm_circum	Forearm circumference in inches
Wrist_circum	Wrist circumference in inches

Note: This dataset is available in the companion directory of the book, in the following directory: “... \Exercises_book_ABME\CH5\MATLAB_BFE\Body_Fat_Estimation_Males_USA.csv.”

TABLE 5.5 For “Body Fat Percentage Target” for the Dataset “Body_Fat_Estimation_Males_USA.csv.”

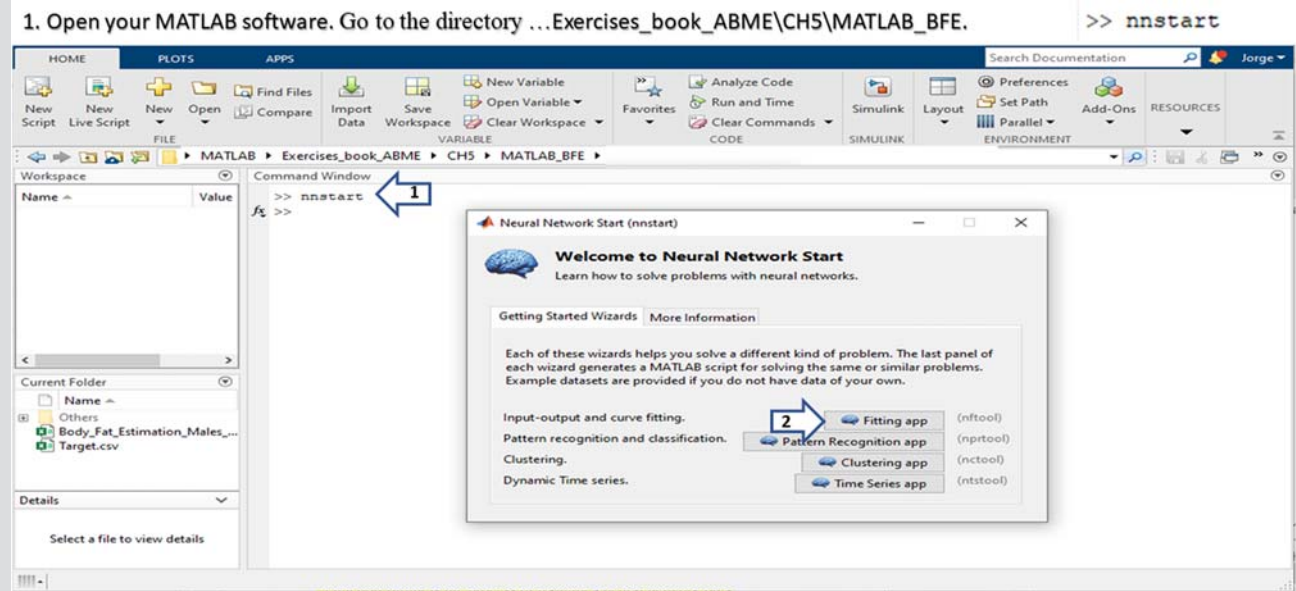
Field	Description * 252 instances all males. Number of fields = 1
Target	Body fat percentage target (decimal)

Note: This dataset is available in the companion directory of the book, in the following directory: “... \Exercises_book_ABME\CH5\MATLAB_BFE\Target.csv.”

Table of slides 5.1 Steps to obtain an AI model “using MATLAB Deep Learning Toolbox for a Feed Forward Neural Network Fitting Apps of a Body Fat Estimation dataset.”

Slide 1 Description Screen figure

1 Open your MATLAB software. Go to the directory “...Exercises_book_ABME\CH5\MATLAB_BFE”. In the command prompt enter the command for Neural Network Starter >> nnstart “MATLAB Deep Learning Toolbox” includes a “Neural Network GUI – nnstart” that opens a window with launch buttons for “Neural Network Fitting,” “Pattern Recognition,” “Clustering” and “Time series tools.” Click the button: “Fitting app (nftool)”



In the command prompt enter the command for Neural Network Starter: MATLAB Deep Learning Toolbox includes a “Neural Network GUI – nnstart” that opens a window with launch buttons for “Neural Network Fitting,” “Pattern Recognition,” “Clustering” and “Time series tools”. Click the button: “Fitting app (nftool)”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

2 In the MATLAB "Neural Fitting" window screen "Neural Network Fitting" is used when here is a need to map a "dataset of numeric inputs" with a numerical target using "Feed Forward Neural", with a specified "Sigmoid hidden neurons" and a "linear output." Click the button "Next"

2. In the MATLAB "Neural Fitting" window screen

The screenshot shows the MATLAB Neural Fitting app interface. The window title is "Neural Fitting (infotool)". The main content area displays a "Welcome to the Neural Network Fitting app." message, followed by an "Introduction" section. The introduction explains that the app solves an input-output fitting problem with a two-layer feed-forward neural network. It provides examples of fitting problems, such as estimating engine emission levels based on fuel consumption and speed, or predicting a patient's bodyfat level based on body measurements. The text also states that the app will help select data, create and train a network, and evaluate its performance using mean square error and regression analysis. A diagram of the neural network architecture is shown, consisting of an Input layer, a Hidden Layer (with sigmoid neurons), and an Output Layer (with linear neurons). The diagram labels the weights (W) and biases (b) for each layer. Below the diagram, the text explains that a two-layer feed-forward network with sigmoid hidden neurons and linear output neurons (ffnnet) can fit multi-dimensional mapping problems arbitrarily well, given consistent data and enough neurons in its hidden layer. It also notes that the network will be trained with Levenberg-Marquardt backpropagation (trainlm), unless there is not enough memory, in which case scaled conjugate gradient backpropagation (trainscg) will be used. At the bottom of the window, there are buttons for "Neural Network Start", "Welcome", "Back", "Next", and "Cancel". A blue arrow points to the "Next" button.

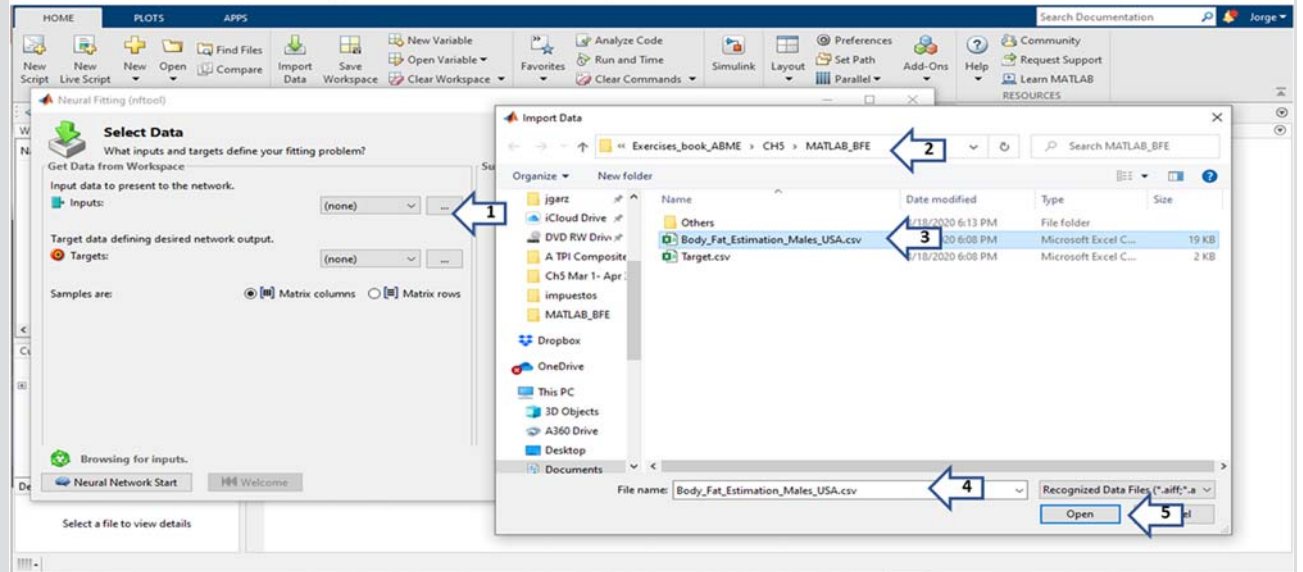
"Neural Network Fitting" is used when here is a need to map a "dataset of numeric inputs" with a numerical target using "Feed Forward Neural", with a specified "Sigmoid hidden neurons" and a "linear output". Click the button "Next"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

3

In the MATLAB "Neural Fitting" window screen: "Select Input Data" Select the button for select the "input dataset ...," select the file "... \Exercises_book_ABME\CH5 \MATLAB_BFE \Body_Fat_Estimation_Males_USA.csv," and click the button "Open"

3. In the MATLAB "Neural Fitting" window screen: "Select Input Data"



Select the button for select the "input dataset ...," select the file "... \Exercises_book_ABME\CH5\MATLAB_BFE\Body_Fat_Estimation_Males_USA.csv", and click the button "Open"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

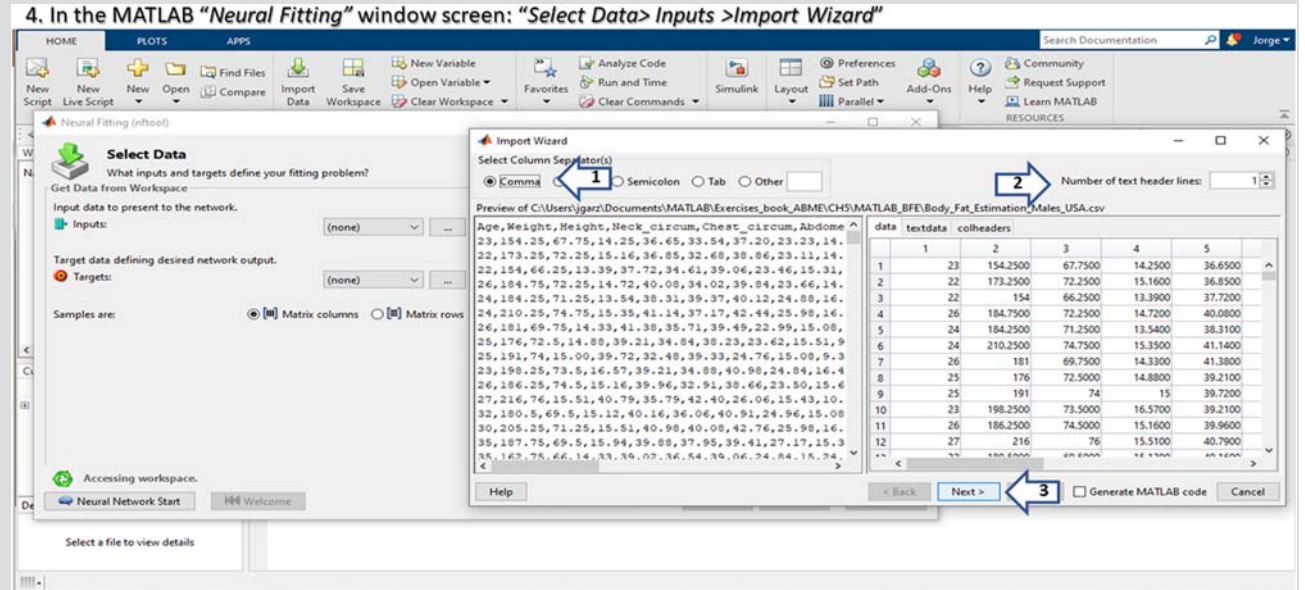
(Continued)

(Continued)

Slide Description

Screen figure

4 In the MATLAB "Neural Fitting" window screen: "Select Data > Inputs > Import Wizard" Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" button



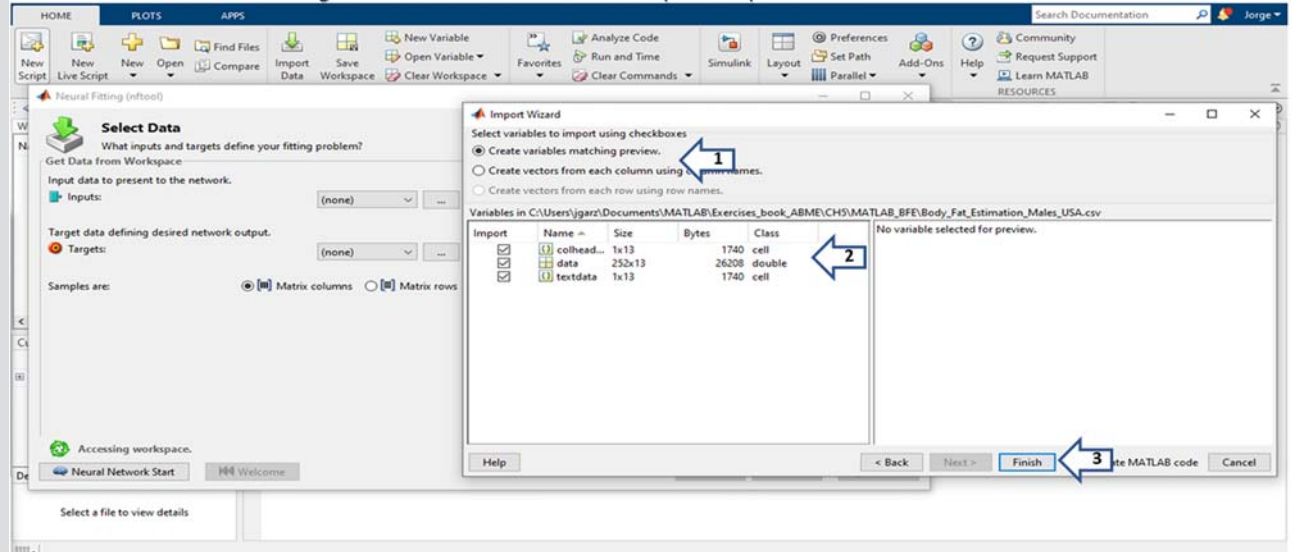
Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" Button

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

5

In the MATLAB “Neural Fitting” window screen: “Select Data > Inputs > Import Wizard” In the section “Select variable to import using checkboxes” activate “Create variables matching preview,” verify the creation of the three variables and click the “Finish” button

5. In the MATLAB “Neural Fitting” window screen: “Select Data>Inputs>Import Wizard”



In the section “Select variable to import using checkboxes” activate “Create variables matching preview”, verify the creation of the 3 variables and click the “Finish” button.

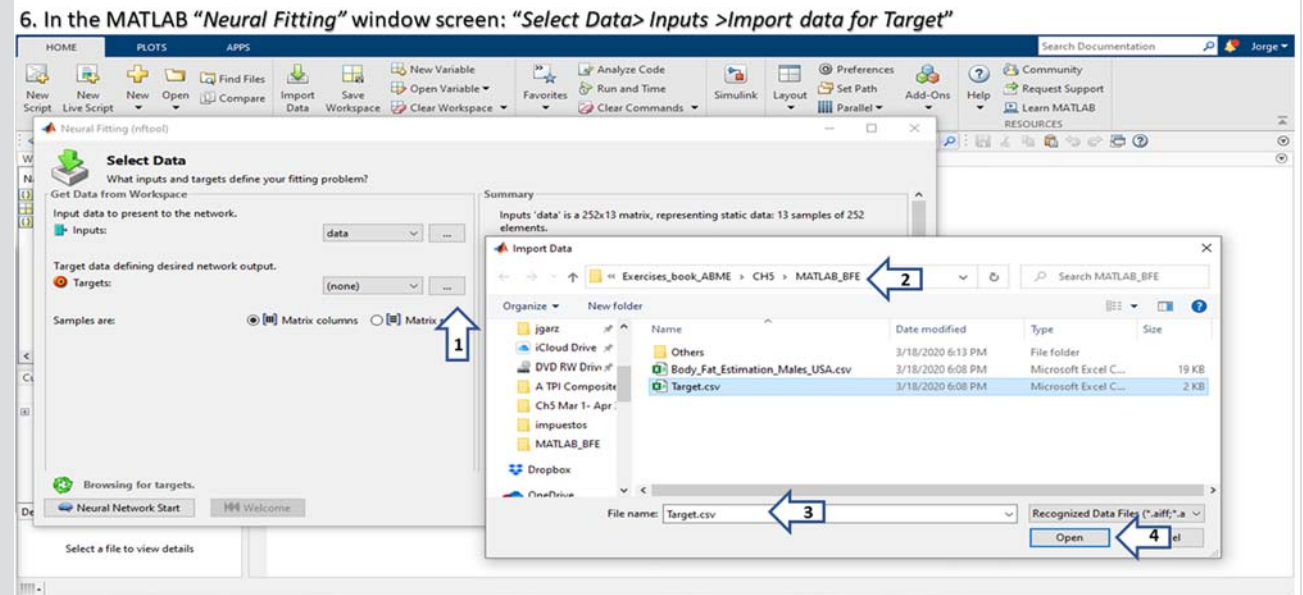
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 6 Description In the MATLAB "Neural Fitting" window screen: "Select Data > Inputs > Import data for Target" Select the button for select the "target data ...," select the file "... \Exercises_book_ABME\CH5 \MATLAB_BFE\Target.csv," and click the button "Open"

Screen figure

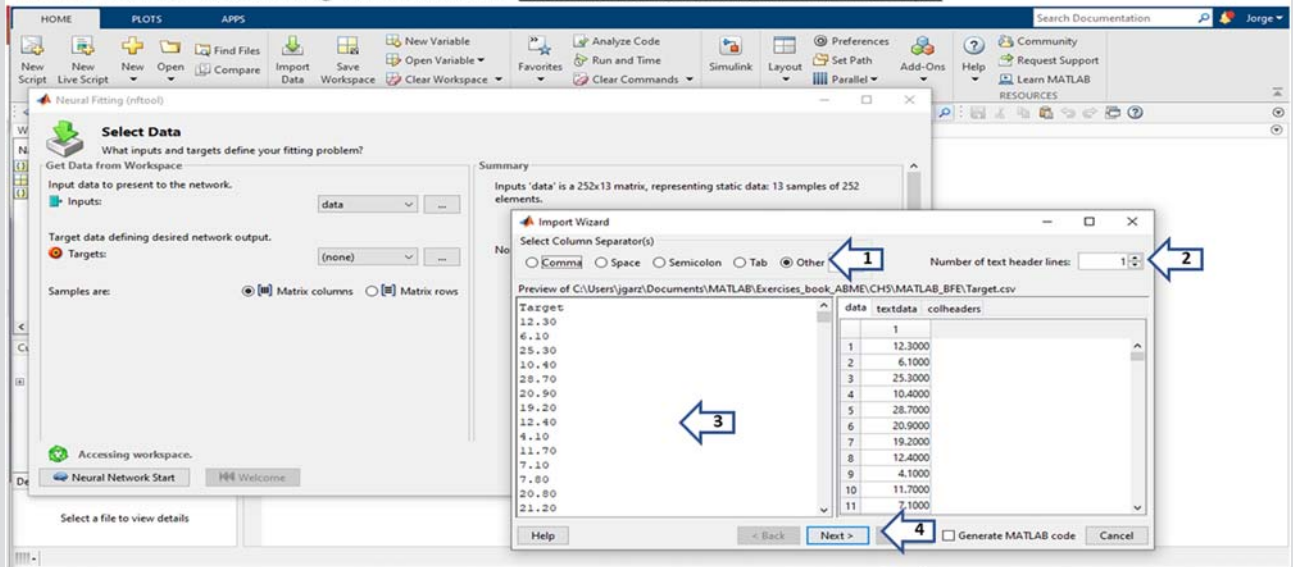


Select the button for select the "target data ...," select the file "... \Exercises_book_ABME\CH5\MATLAB_BFE\Target.csv", and click the button "Open"

7

In the MATLAB "Neural Fitting" window screen: "Select Data > Inputs > Import data for Target" Select "Other" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" button

7. In the MATLAB "Neural Fitting" window screen: "Select Data>Inputs>Import data for Target"



Select "Other" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" Button"

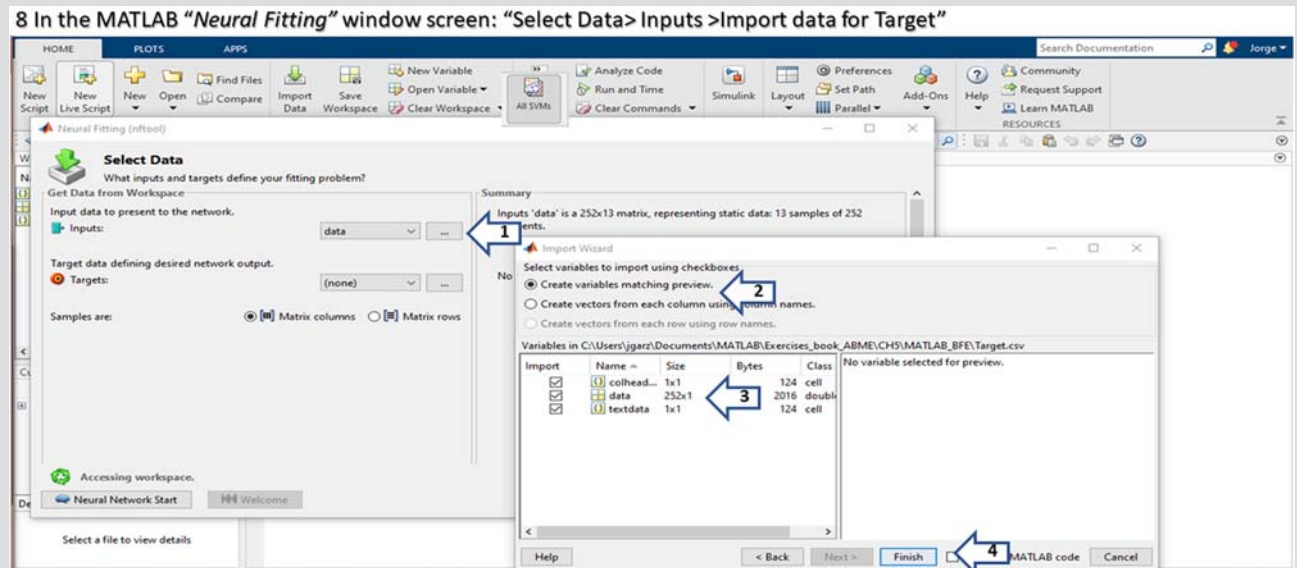
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 8 Description In the MATLAB "Neural Fitting" window screen: "Select Data > Inputs > Import data for Target" In the section "Select variable to import using checkboxes" activate "Create variables matching preview," verify the creation of the three variables and click the "Finish" button

Screen figure

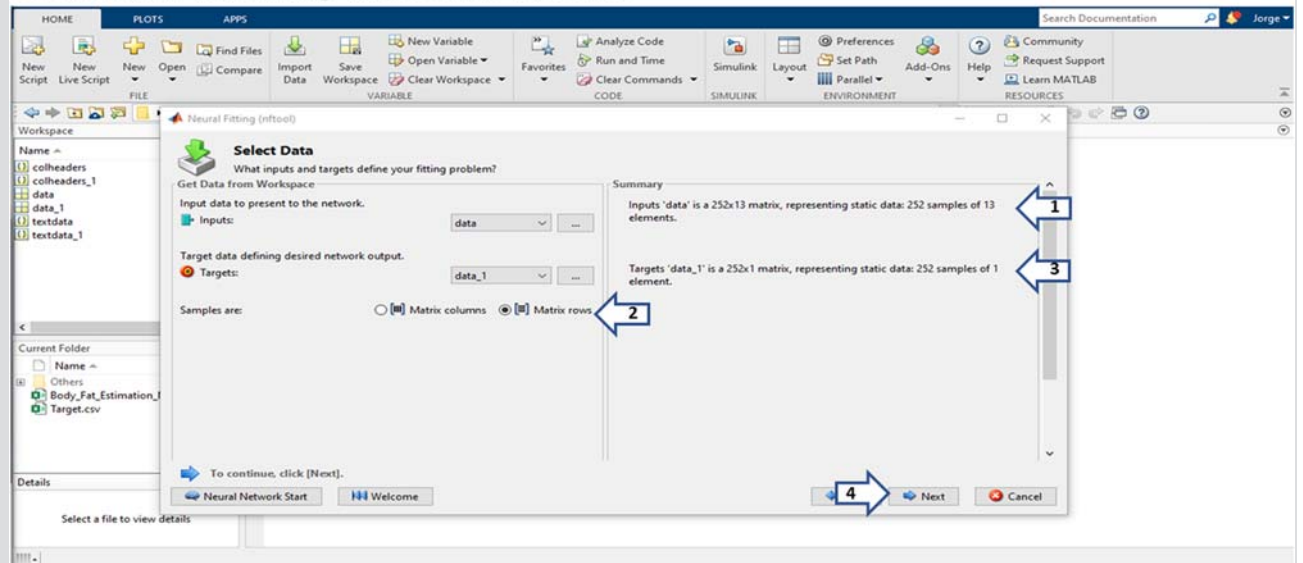


In the section "Select variable to import using checkboxes" activate "Create variables matching preview", verify the creation of the 3 variables and click the "Finish" button.

9

In the MATLAB "Neural Fitting" window screen: "Select Data"
In the "Summary" section verify that the "input data is 252×13 matrix," specify the target as "Matrix rows," check that the "Targets data is a 252×1 matrix," and click in the button "Next"

9. In the MATLAB "Neural Fitting" window screen: "Select Data"



In the "Summary" section verify that the "input data is 252×13 matrix", specify the target as "Matrix rows", check that the "Targets data is a 252×1 matrix", and click in the button "Next"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

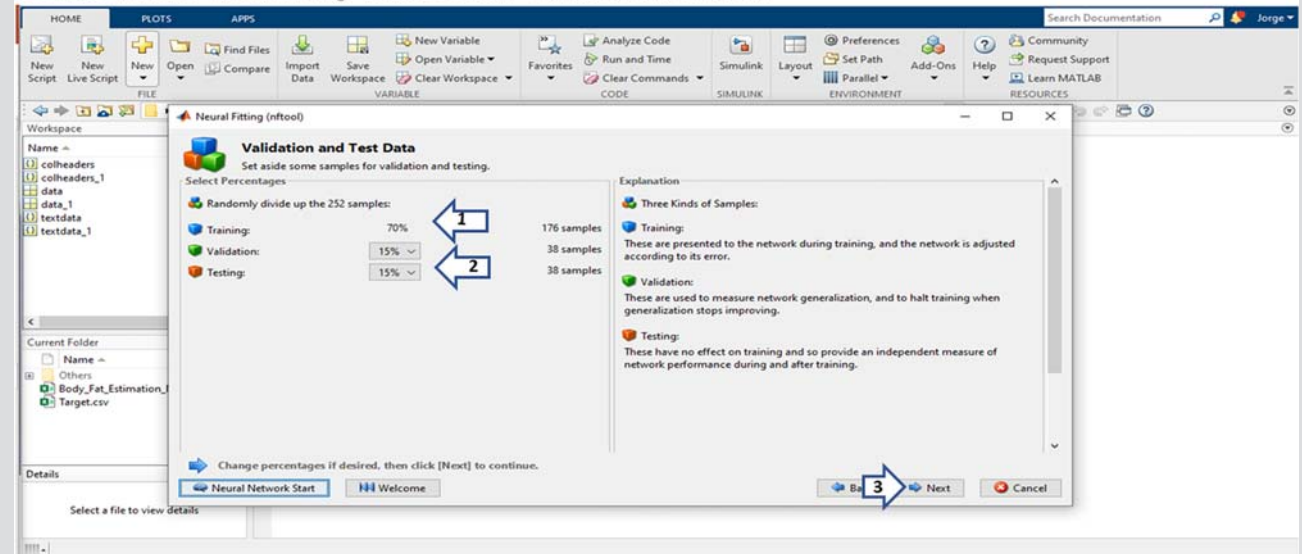
(Continued)

(Continued)

Slide 10 Description In the MATLAB "Neural Fitting" window screen: "Validation and Test Data" In the section "Select Percentages" verify that: "Training = 70%," "Validation = 15%" and "Testing = 15%," then click in the button "Next"

Screen figure

10. In the MATLAB "Neural Fitting" window screen: "Validation and Test Data"



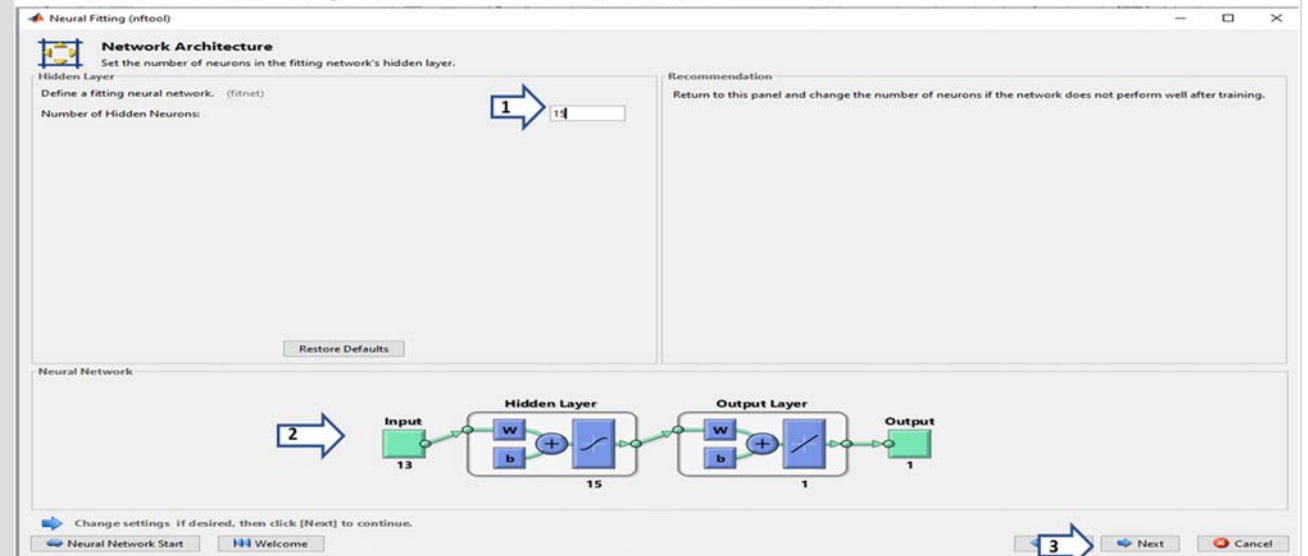
In the section "Select Percentages" verify that: "Training=70%," "Validation=15%" and "Testing=15%," then click in the button "Next".

11

In the MATLAB "Neural Fitting" window screen: "Network Architecture"

In the "Hidden Layer" section, define a "Fitting Neural Network with 15 hidden neurons," in the "Neural Network" section verify that the "ANN" is defined with: "input of 13 variables," "Hidden Layers = 1 with 15 neurons," and "Output Layer of 1 variable," and 1 output. Then click in the button "Next"

11 In the MATLAB "Neural Fitting" window screen: "Network Architecture"



In the "Hidden Layer" section, define a "Fitting Neural Network with 15 hidden neurons", in the "Neural Network" section verify that the "ANN" is defined with: "input of 13 variables", "Hidden Layers=1 with 15 neurons ", and "Output Layer of 1 variable", and 1 output. Then click in the button "Next".

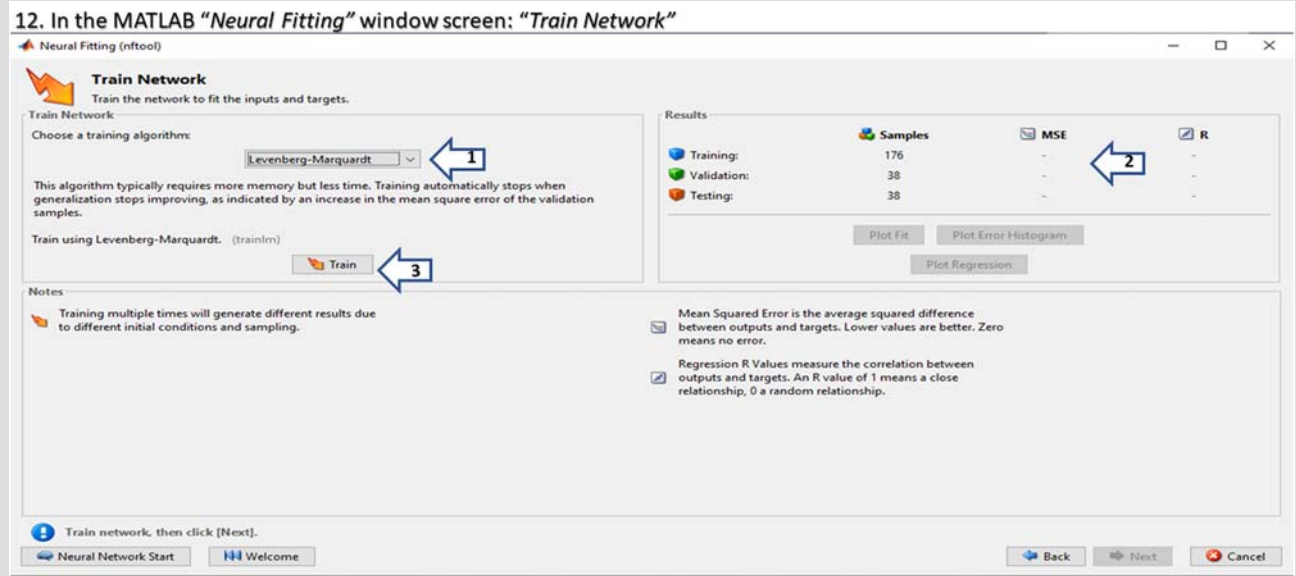
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 12 Description In the MATLAB “Neural Fitting” window screen: “Train Network” In the “Train Network” section choose: “Levenberg–Marquart,” this algorithm is ideal for small input data, usually require more memory but less time, and the training stop automatically when it is not improving based on an increase in the “Mean Square Error (MSE)” of the validation samples. Then click in the “Train” button Note: “Levenberg–Marquardt backpropagation” is an optimization method that update weight and bias

Screen figure

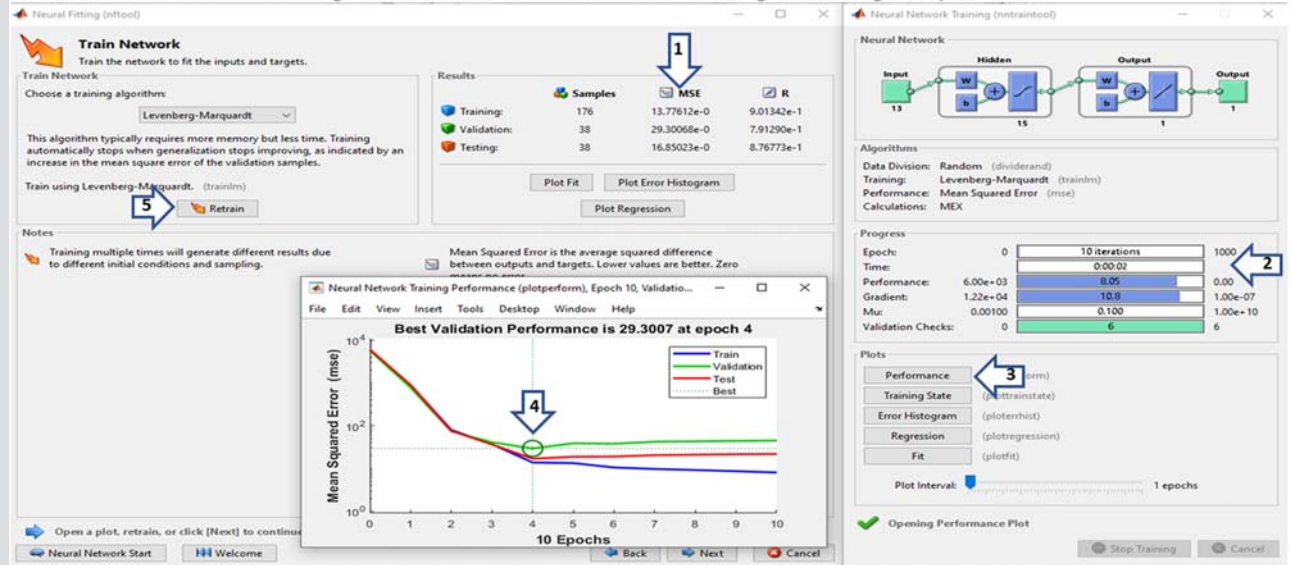


In the “Train Network” section choose: “Levenberg-Marquart”, this algorithm is ideal for small input data, usually require more memory but less time, and the training stop automatically when it is not improving based on an increase in the “Mean Square Error (MSE)” of the validation samples. Then click in the “Train” button.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

In the MATLAB “Neural Fitting” window screen: “Neural Network Training—Levenberg—Marquart” Observe “Mean Square Error (MSE)” for Training, Validation, and Testing, check the “Progress” to obtain the result: “Epoch,” “Time,” “Performance,” “Gradient,” “Mu,” and “Validation Checks.” In the “Plots” section click “Performance” button and observe the “Best Validation Performance as 29.3007 at epoch 4.” (Note: The values can be different, depend on the initial random value). Click the “Retrain” button for lower MSE values”

13. In the MATLAB “Neural Fitting” window screen: “Neural Network Training - Levenberg-Marquart”



Observe “Mean Square Error (MSE)” for Training, Validation and Testing, check the “Progress” to obtain the result: “Epoch,” “Time,” “Performance,” “Gradient,” “Mu” and “Validation Checks”. In the “Plots” section click “Performance” Button, and observe “Best Validation Performance as 29.3007 at epoch 4”. (Note: The values can be different, depend of the initial random value). Click the “Retrain” button for lower MSE values”

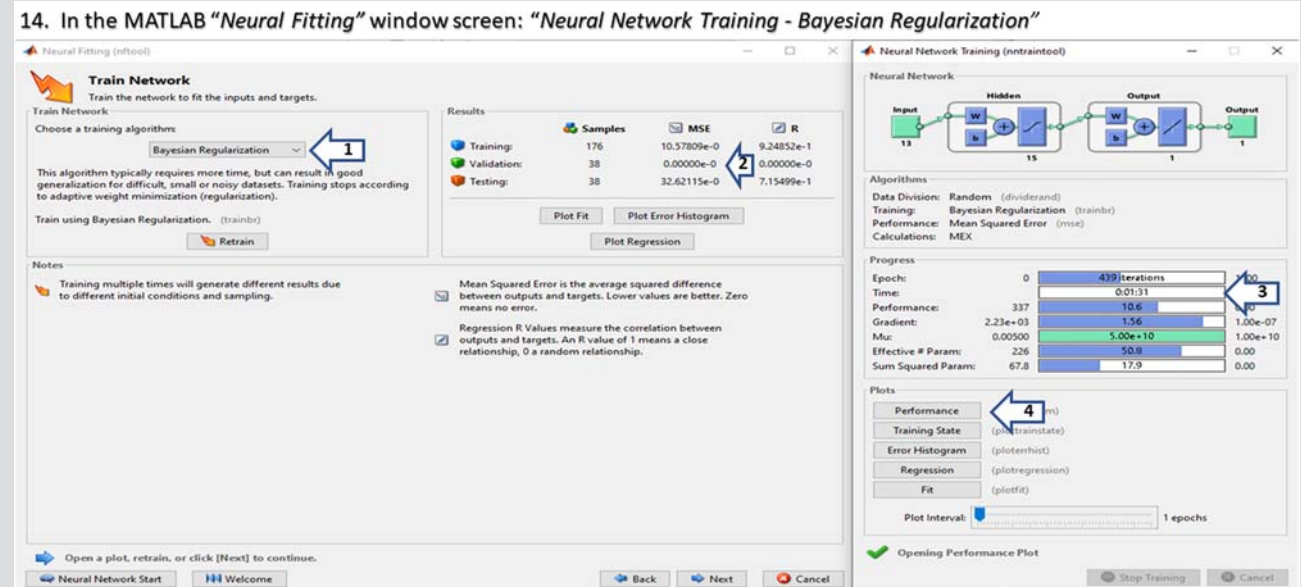
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ujioa

(Continued)

(Continued)

Slide 14 Description In the MATLAB “Neural Fitting” window screen: “Neural Network Training—Bayesian Regularization” In the “Train Network” section choose: “Bayesian Regularization,” this algorithm takes more time, but usually give better results for some dataset and training stop when a minimum is reached on “Regularization.” Check the “Progress” section to obtain the result. In the “Plots” section click “Performance” button and observe “Best Validation Performance as 29.3007 at epoch 4”

Screen figure



In the “Train Network” section choose: “Bayesian Regularization”, this algorithm takes more time, but usually give better results for some dataset and training stop when a minimum is reached on “Regularization”. Check the “Progress” section to obtain the result. In the “Plots” section click “Performance” Button, and observe “Best Validation Performance as 29.3007 at epoch 4”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

In the MATLAB "Neural Fitting" window screen: "Neural Network Training—Scaled Conjugate Gradient" In the "Train Network" section choose: "Scaled Conjugate Gradient," this algorithm takes less memory, more time, usually is used in larger dataset and training stop when a minimum is reached on "Regularization." Check the "Progress" section to obtain the result, in the "Plots" section click "Performance" button and observe "Best Validation Performance as 22.0267 at epoch 35." Click "Next" button

15. In the MATLAB "Neural Fitting" window screen: "Neural Network Training - Scaled Conjugate Gradient"

The screenshot displays the MATLAB Neural Fitting interface. The main window is titled "Neural Network Training - Scaled Conjugate Gradient".

Train Network Section: The "Choose a training algorithm" dropdown is set to "Scaled Conjugate Gradient". A "Retrain" button is visible.

Results Table:

	Samples	MSE	R
Training:	176	13.97987e-0	9.06887e-1
Validation:	38	22.02665e-0	7.19859e-1
Testing:	38	31.00556e-0	7.39805e-1

Neural Network Training Performance (plotperform), Epoch 41, Validation...

Best Validation Performance is 22.0267 at epoch 35

The plot shows Mean Squared Error (mse) on a logarithmic scale (10⁰ to 10³) versus Epochs (0 to 40). Three lines are shown: Train (blue), Validation (green), and Best (red). The validation error reaches its minimum of 22.0267 at epoch 35.

Progress Section: A bar chart shows the progress of training. The "Performance" bar is highlighted in blue, indicating the current state. The "Epochs" bar shows 41 iterations out of 1000. The "Time" bar shows 0.0007. The "Gradient" bar shows 1.51e+03. The "Validation Checks" bar shows 6 out of 6.

Plots Section: The "Performance" button is selected.

Buttons: "Next" and "Cancel" buttons are visible at the bottom of the plot window.

In the "Train Network" section choose: "Scaled Conjugate Gradient", this algorithm takes less memory, more time, usually is used in larger dataset and training stop when a minimum is reached on "Regularization". Check the "Progress" section to obtain the result, in the "Plots" section click "Performance" Button, and observe "Best Validation Performance as 22.0267 at epoch 35". Click "Next" button.

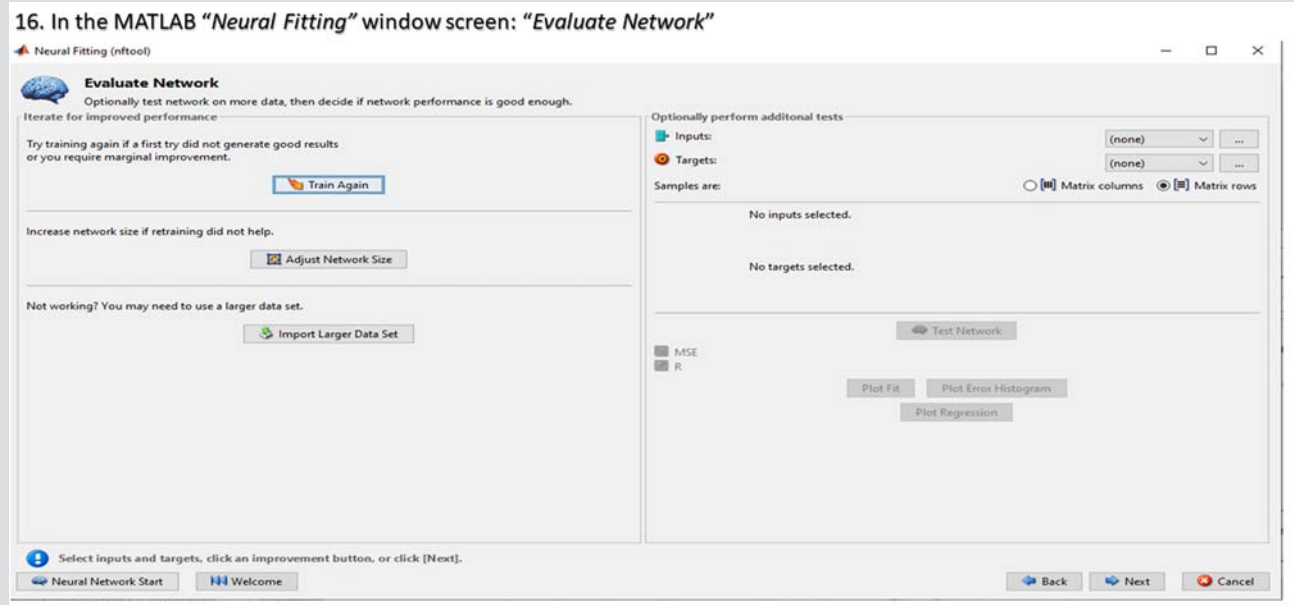
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 16 Description In the MATLAB "Neural Fitting" window screen: "Evaluate Network" Feel free to reevaluate the network to try to find better values with: "Train Again," "Adjust Network Size," "Import Larger Dataset," "Optionally perform additional test," etc. Click the "Next" button

Screen figure

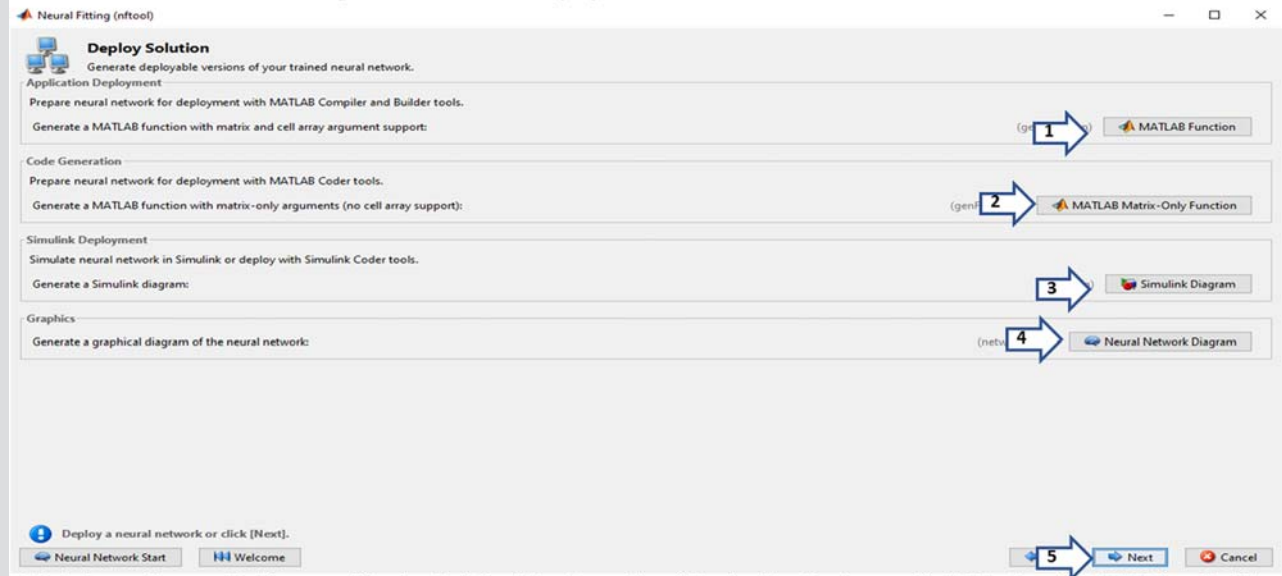


Feel free to re-evaluate the network to try to find better values with: "Train Again", "Adjust Network Size", "Import Larger Data Set", "Optionally perform additional test", etc. Click the "Next" Button.

17

In the MATLAB "Neural Fitting" window screen: "Deploy Solution"
Select the option needed for your "Deploy Solution": In the section "Application Deployment" click button "MATLAB function," in "Code Generation" click button "MATLAB Matrix-Only Function," in "Simulink Deployment" click "Simulink Diagram" and in "Graphics" click button "Neural Network Diagram." Click "Next" button

17. In the MATLAB "Neural Fitting" window screen: "Deploy Solution"



Select the option needed for your "Deploy Solution": In the section "Application Deployment" click button "MATLAB function", in "Code Generation" click button "MATLAB Matrix-Only Function", in "Simulink Deployment" click "Simulink Diagram" and in "Graphics" click button "Neural Network Diagram". Click "Next" Button.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

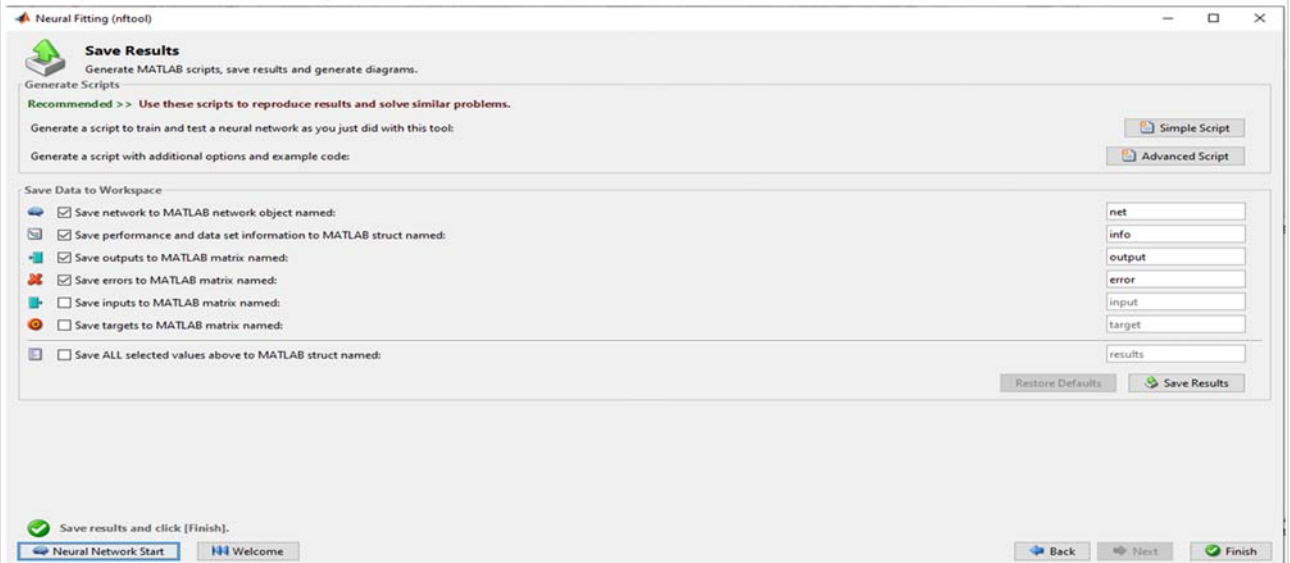
(Continued)

(Continued)

Slide 18 Description In the MATLAB "Neural Fitting" window screen: "Save Results" In the section "Generate Scripts" click the buttons: "Simple Script" and "Advanced Script." In the section "Save Data to Workspace" click button "Save Results." Finally click on the button "Finish," and its confirmation dialog

Screen figure

18. In the MATLAB "Neural Fitting" window screen: "Save Results"

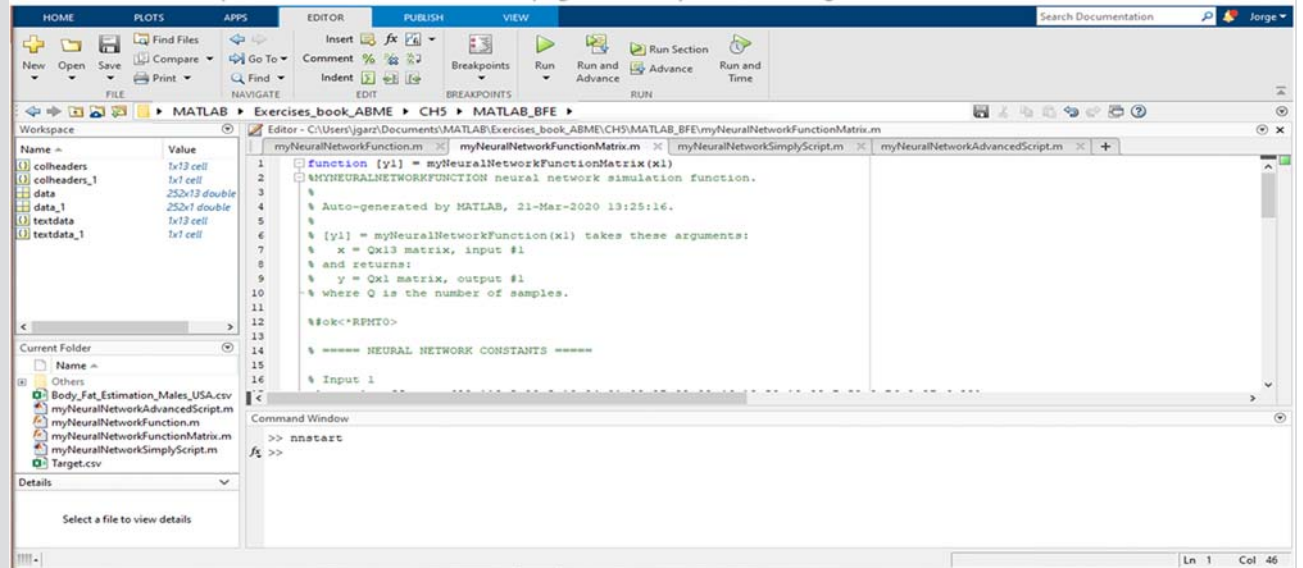


In the section "Generate Scripts" click the buttons: "Simple Script" and "Advanced Script". In the section "Save Data to Workspace" click button "Save Results". Finally click on the button "Finish", and its confirmation dialog.

19

In MATLAB workspace save the functions and script generated by "Neural Fitting"
Save the two functions generated by MATLAB "Neural Fitting" as:
"myNeuralNetworkFunction.m" and
"myNeuralNetworkFunctionMatrix.m."
Save also the two scripts as:
"myNeuralNetworkSimplyScript.m"
and
"myNeuralNetworkAdvancedScript.m"

19. In MATLAB workspace save the functions and script generated by "Neural Fitting"



Save the two functions generated by MATLAB "Neural Fitting" as: "myNeuralNetworkFunction.m" and "myNeuralNetworkFunctionMatrix.m". Save also the two scripts as: "myNeuralNetworkSimplyScript.m" and "myNeuralNetworkAdvancedScript.m"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

20 In MATLAB workspace testing with new data the "myNeuralNetworkFunctionMatrix function" generated by "Neural Fitting" To evaluate a new data as a matrix, load the file "x1.csv," show its contents with the 13 attributes for the 2 new records, call the function created by "Neural Fitting" and obtain the results for "Body Fat Percentage." Where the first record with "Body Fat percentage" of 30.01% is in the "Body Fat Ranges" of "obese," and the second of 25.53% is ranged between "Overfat" and "Obese"

20. In MATLAB workspace testing with new data the "myNeuralNetworkFunctionMatrix function" generated by "Neural Fitting"

The screenshot shows the MATLAB environment with the following components:

- Workspace:** Lists variables including colheaders, data, x1, and y1.
- Command Window:** Shows the execution of the following commands:

```
>> load('x1.csv')
>> x1
x1 =
    74.0000    207.5000    70.0000    16.0600    44.2500    42.7200    42.1700    23.3500    16.6100    9.6900    13.2700    11.8100    8.2300
    72.0000    190.7500    70.5000    15.3100    42.6400    39.8800    38.5000    22.0500    16.3800    8.9400    12.0100    11.5700    7.8000

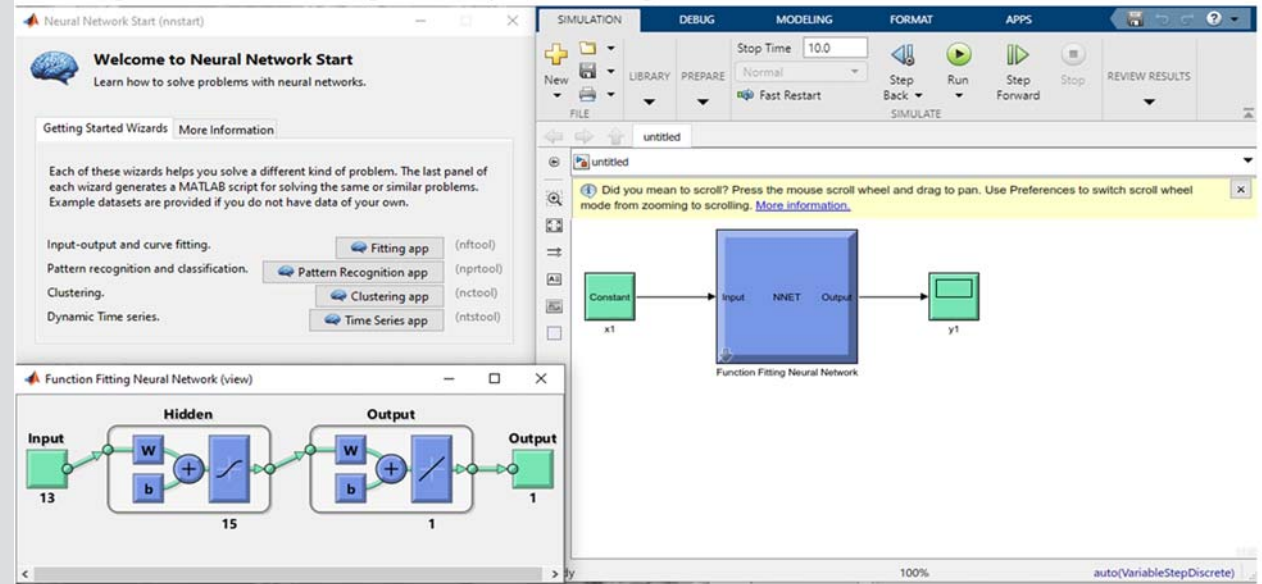
>> [y1] = myNeuralNetworkFunction(x1)
y1 =
    30.0117
    25.5331
```
- Diagram:** A "Function Fitting Neural Network (view)" diagram showing an input layer with 13 nodes, a hidden layer with 15 nodes, and an output layer with 1 node. The diagram includes weights (W) and biases (b) for each layer.

To evaluate a new data as a matrix, load the file "x1.csv", show its contents with the 13 attributes for the 2 new records, call the function created by "Neural Fitting" and obtain the results for "Body Fat Percentage". Where the first record with "Body Fat percentage" of 30.01% is in the "Body Fat Ranges" of "obese", and the second of 25.53% is ranged between "Overfat" and "Obese"

21

Closing MATLAB tools and Plots generated by “Neural Network Fitting”
Close without saving the “*Simulink diagram for the ANN*” and the graphical ANN generated by the “*Neural Network-Fitting App*”

21. Closing MATLAB tools and Plots generated by “Neural Fitting”



Close without saving the “*Simulink diagram for the ANN*” and the graphical ANN generated by the “*Neural Network – Fitting App*”.

Conclusions

Obtaining an AI model for FFNN fitting apps of a body fat estimation dataset allow the prediction of the “body fat percentage” and to classify into specific “body fat ranges.”

Recommendation

- A larger dataset will help to obtain with more precision the “body fat percentage” to locate the “body fat ranges” using the “AI FFNN Model.”

5.3.2 Research 52 Neural Network for clustering based on “Self-Organizing MAP* through a Shallow Neural Network” to analyze Surface Electromyography signals

Note*: “Self-Organizing Map (SOM) are also known as Korhonen Network (KN) to be studied in the section of “Recurrent Neural Network section 6.2”.

5.3.2.1 Case for research

“Obtain an AI graphic representation model with MATLAB Deep Learning Toolbox using a Neural Network for clustering based on Self-Organizing MAP (SOM) through a Shallow Neural Network to visually analyze Surface Electromyography (sEMG) signals in diabetes type 2 patient.”

5.3.2.2 General objective

Apply “MATLAB Deep Learning Toolbox” to define, build, train, and deploy a “Self-Organizing MAP as a Shallow Neural Network” to visually analyze two consecutive datasets of diabetic type 2 patients, from the right limb soleus muscle obtained by sEMG signals on two consecutive monthly visits*.”

Note*: This research shows how it is possible to obtain a “visual correlation between two consecutive monthly visits to obtain surface electromyography signals from a diabetes type 2 patient.” Note: This is an analysis based on “basic visual feedback for the representation of 2 on two consecutive monthly visits,” with a purpose to introduce “self-organizing map (SOM)” that consists of a competitive layer as opposed to “error-correction learning” (instead of the traditional backpropagation with gradient descent) to classify a dataset of vectors with any number of dimensions into as many classes as the layer has neurons. The neurons are arranged in a two-dimensional (2D) topology, which allows the layer to

form a representation of the distribution and a 2D approximation of the topology of the dataset.

5.3.2.3 Specific objectives

- Load the “Visit_Measurements.csv” dataset that contains two sEMG vectors, one for each visit. The normalized SEMG signals were obtained from the right limb soleus muscle during two consecutive monthly visits. The input matrix will be data to be analyze by the “Self-Organizing MAP through a Shallow Neural Network.”
- Select the optimal number of “Self-Organizing Map (SOM) layer” for the best performance in the “Neural Network Clustering.”
- Train the “Self-Organizing MAP through a Shallow Neural Network using a batch SOM algorithm” for the “Neural Network Clustering.”
- Visualize the cluster behaviors using plots for:
 - “SOM Topology” is defined as 10×10 for 100 neurons.
 - “SOM Sample Hits” to find neuron locations, maximum number of hits associated with each neuron, and their “cluster centers.”
 - “SOM Input Planes” show the visualization for the two vector inputs as “Weight Planes.”
 - “SOM Neighbor Distance” shows with lines the distance between neighboring neurons.
- Deploy the MATLAB “Neural Network Clustering” “AI model as a MATLAB function and as a “Neural Network Advanced Script.”
- Test the MATLAB “Neural Network Advanced Script” model with a matrix based on two sEMG normalized signals vectors inputs from two datasets of “healthy patient right limb Soleus muscle” obtained in two consecutive monthly visits.”

5.3.2.4 Background for “sEMG”

One of the most common biomedical signals is obtained as an “Surface Electromyography (sEMG) signal”; this is an external measure of the electrical currents generated during skeletal muscles activity by its contraction, representing neuromuscular activities. This instrument produces a record of signal behaviors called an “electromyogram.” The nervous system controls the muscle activity under contraction and relaxation and is dependent on the anatomical and physiological properties of the muscle, the “sEMG measures the electrical potential generated from some group muscle fibers or muscle cells producing a characteristic biphasic waveform.” The “skeletal

muscles” tissue is attached to the bone and its contraction is responsible for supporting and moving the skeleton. The contraction of the “*skeletal muscle is initiated by impulses from the neurons to the muscle and is usually under voluntary control.*” The skeletal muscle fibers have many neurons called “*motor neurons*”; one motor neuron usually supplies stimulation to many muscle fibers. In addition, a group of motor units often works together to coordinate the contraction of a single muscle [19].

5.3.2.5 Background for “Diabetes Mellitus”

“*Diabetes Mellitus (DM)*” is a disease that occurs when blood glucose is too high. “*Blood glucose*” is the main source of energy and comes from food. Insulin, a hormone made by the pancreas, helps glucose from food get into your cells to be used for energy. There are two main types of diabetes: “*type 1 DM*” and “*type 2 DM*” [20]:

- “*Type 1 DM*” occurs because the insulin-producing cells of the “*pancreas (beta cells)*” are damaged. In “*type 1 DM, the pancreas makes little or no insulin,*” so sugar cannot get into the body’s cells for use as energy. “*People with type 1 diabetes must use insulin injections to control their blood glucose.*”
- “*Type 2 DM*” occurs when the pancreas makes insulin, but it either does not produce enough, or the insulin does not work properly. Nine out of 10 people with

diabetes have type 2. This type occurs most often in people who are over 40 years old but can occur even in childhood if there are risk factors present. “*Type 2 diabetes mellitus affects muscles, typically in the lower extremities, by impairing blood flow at the macrovascular and microvascular levels*” [81] The “*soleus muscle*” is the plantar flexor muscle of the ankle. It can exert powerful forces onto the ankle joint. It is located on the back of the lower leg and originates at the posterior (rear) aspect of the fibular head and the medial border of the tibial shaft. It is frequently affected in patients with “*type 2 DM*” [29].

5.3.2.6 Dataset

The dataset “*Visit_Measurements.csv*” is based on normalized sEMG signals from the right soleus muscles in two consecutive monthly visits as a matrix of two vectors, as indicated in Table 5.6.

5.3.2.7 Procedure

The steps to obtain an AI model “*using MATLAB Deep Learning Toolbox using a Neural Network for clustering based on “Self-Organizing MAP through a Shallow Neural Network”* to visually analyze “*Surface Electromyography signals*” are summarized in Table of slides 5.2, and each step of the example is visually explained using screen sequences with instructions in easy to follow screen figures.

TABLE 5.6 Dataset “*Visit_Measurements.csv*” fields and descriptions.

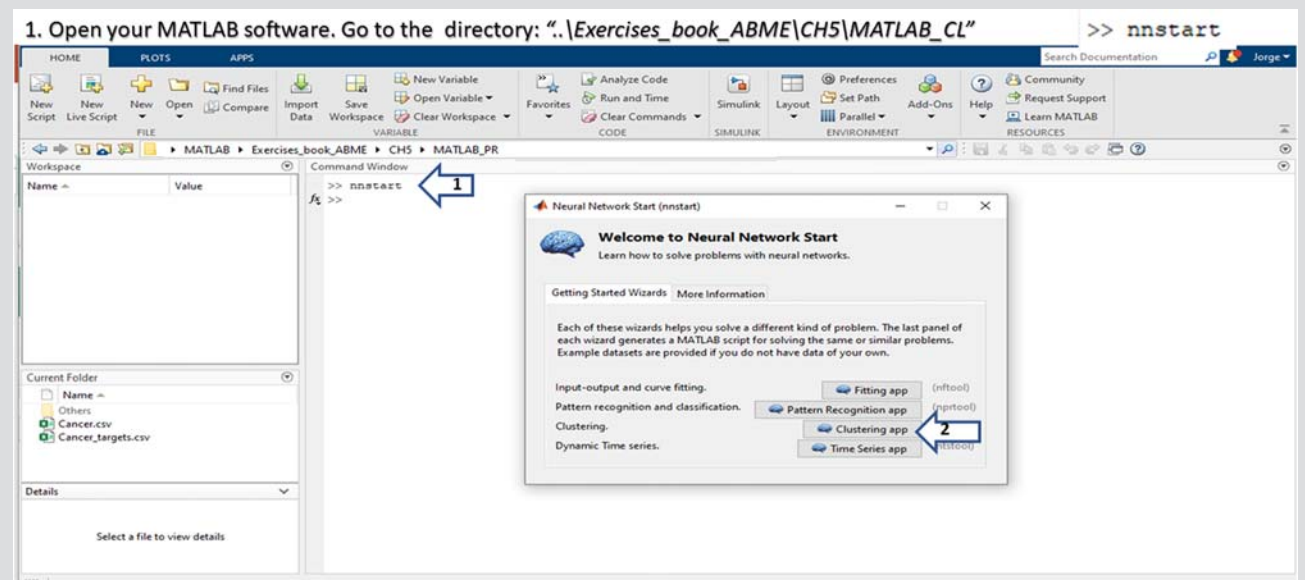
Field	Description * 1000 instances of sEMG normalized values. Number of fields = 2
Visit1	Lectures values in 1 day first visit of a “ <i>type 2 DM</i> ” patient (normalized decimal)
Visit2	Lectures values in another day second visit of the same “ <i>type 2 DM</i> ” patient a month later (normalized decimal)

Note: This dataset is available in the companion directory of the book, in the following directory: “*.. \Exercises_book_ABME\CH5\MATLAB_CL \Visit_Measurements.csv.*”

Table of slides 5.2 Steps for MATLAB Deep Learning Toolbox using a Neural Network for clustering based on “Self-Organizing MAP through a Shallow Neural Network” to analyze Surface Electromyography signals.

Slide 1 Description Screen figure

1 Open your MATLAB software. Go to the directory: “... \Exercises_book_ABME\CH5 \MATLAB_CL”. In the command prompt enter the command for Neural Network Starter >> *nnstart* “MATLAB Deep Learning Toolbox” includes a “Neural Network GUI—*nnstart*” that opens a window with launch buttons for “Neural Network Fitting,” “Pattern Recognition,” “Clustering,” and “Time series tools.” Click the button: “Pattern Recognition App (*nprtool*)”



In the command prompt enter the command for Neural Network Starter: “MATLAB Deep Learning Toolbox™” includes a “Neural Network GUI – *nnstart*” that opens a window with launch buttons for “Neural Network Fitting”, “Pattern Recognition”, “Clustering” and “Time series tools”. Click the button: “Clustering App (*nctool*)”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

2

In the MATLAB "Neural Network Clustering" window screen "Neural Network Clustering" is used when there is a need to group a "dataset of numeric inputs" by similarity using a "Self-Organizing Map" algorithm, that consists of a competitive layer that can classify vectors of any number of dimensions into as many classes at the layers has neurons, then it can represent the distribution using a 2D approximation. Click the button "Next"

2. In the MATLAB "Neural Network Clustering" window screen

The screenshot shows the MATLAB environment with the Neural Network Clustering app open. The app window displays a welcome message and an introduction to the Self-Organizing Map (SOM) algorithm. A diagram illustrates the SOM layer architecture, showing an input vector being processed by a layer of neurons (represented by a box labeled 'SOM Layer' with weights 'W' and a bias '+') to produce an output. The text explains that the SOM layer consists of a competitive layer that can classify a dataset of vectors into as many classes as the layer has neurons, arranged in a 2D topology. The network is trained using the SOM batch algorithm (trainbs, learnbsomb). At the bottom of the window, there are buttons for 'Neural Network Start', 'Welcome', 'Back', 'Next', and 'Cancel'. The 'Next' button is highlighted, indicating the next step in the process.

"Neural Network Clustering" is used when there is a need to group a "dataset of numeric inputs" by similarity using a "Self-Organizing Map" algorithm, that consists of a competitive layer that can classify vectors of any number of dimensions into as many classes at the layers has neurons, then it can represent the distribution using a two dimensional approximation. Click the button "Next"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide	Description	Screen figure
3	<p>In the MATLAB "Neural Network Clustering" window screen: "Select input Data" Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" button</p>	<p>3. In the MATLAB "Neural Network Clustering" window screen: "Select input Data"</p> <p>Click on the button for select the "input dataset ...", select the file "...\\Exercises_book_ABME\\CH5\\MATLAB_CL\\Visit_Measurements.csv", and click the button "Open"</p> <p>From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

4

In the MATLAB "Neural Networking Clustering" window screen: "Select Data > Inputs > Import Wizard" Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" button

4. In the MATLAB "Neural Networking Clustering" window screen: "Select Data > Inputs > Import Wizard"

The screenshot shows the MATLAB 'Import Wizard' dialog box. The 'Select Column Separator(s)' section has 'Comma' selected. The 'Number of text header lines' is set to 1. The 'Next >' button is highlighted with a blue arrow labeled '3'. A preview table shows data with 2 columns.

	1	2
1	0.1422	0.4273
2	0.6405	0.5203
3	0.7148	0.3452
4	0.3394	0.2514
5	0.8307	0.5442
6	0.6484	0.7350
7	0.6726	0.5530
8	0.4709	0.4969
9	0.6818	0.6869
10	0.2489	0.1988
11	0.8382	0.9272

Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" Button"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide	Description	Screen figure																				
5	<p>In the MATLAB “Neural Network Clustering” window screen: “Select Data > Inputs > Import Wizard”</p> <p>In the section “Select variable to import using checkboxes” activate “Create variables matching preview,” verify the creation of the three variables and click the “Finish” button</p>	<p>5. In the MATLAB “Neural Network Clustering” window screen: “Select Data > Inputs > Import Wizard”</p> <p>Summary: No inputs selected.</p> <p>Select variables to import using checkboxes:</p> <ul style="list-style-type: none"><input checked="" type="radio"/> Create variables matching preview.<input type="radio"/> Create vectors from each column using column names.<input type="radio"/> Create vectors from each row using row names. <p>Variables in C:\Users\jgarza\Documents\MATLAB\Exercises_book_ABME\CH5\MATLAB_CL\Visit_Measurements.csv</p> <table border="1"><thead><tr><th>Import</th><th>Name</th><th>Size</th><th>Bytes</th><th>Class</th></tr></thead><tbody><tr><td><input checked="" type="checkbox"/></td><td>colhead...</td><td>1x2</td><td>232</td><td>cell</td></tr><tr><td><input checked="" type="checkbox"/></td><td>data</td><td>1000x2</td><td>16000</td><td>doub</td></tr><tr><td><input checked="" type="checkbox"/></td><td>textdata</td><td>1x2</td><td>232</td><td>cell</td></tr></tbody></table> <p>Buttons: < Back, Next >, Finish, Cancel</p>	Import	Name	Size	Bytes	Class	<input checked="" type="checkbox"/>	colhead...	1x2	232	cell	<input checked="" type="checkbox"/>	data	1000x2	16000	doub	<input checked="" type="checkbox"/>	textdata	1x2	232	cell
Import	Name	Size	Bytes	Class																		
<input checked="" type="checkbox"/>	colhead...	1x2	232	cell																		
<input checked="" type="checkbox"/>	data	1000x2	16000	doub																		
<input checked="" type="checkbox"/>	textdata	1x2	232	cell																		

In the section “Select variable to import using checkboxes” activate “Create variables matching preview”, verify the creation of the 3 variables, and click the “Finish” button.

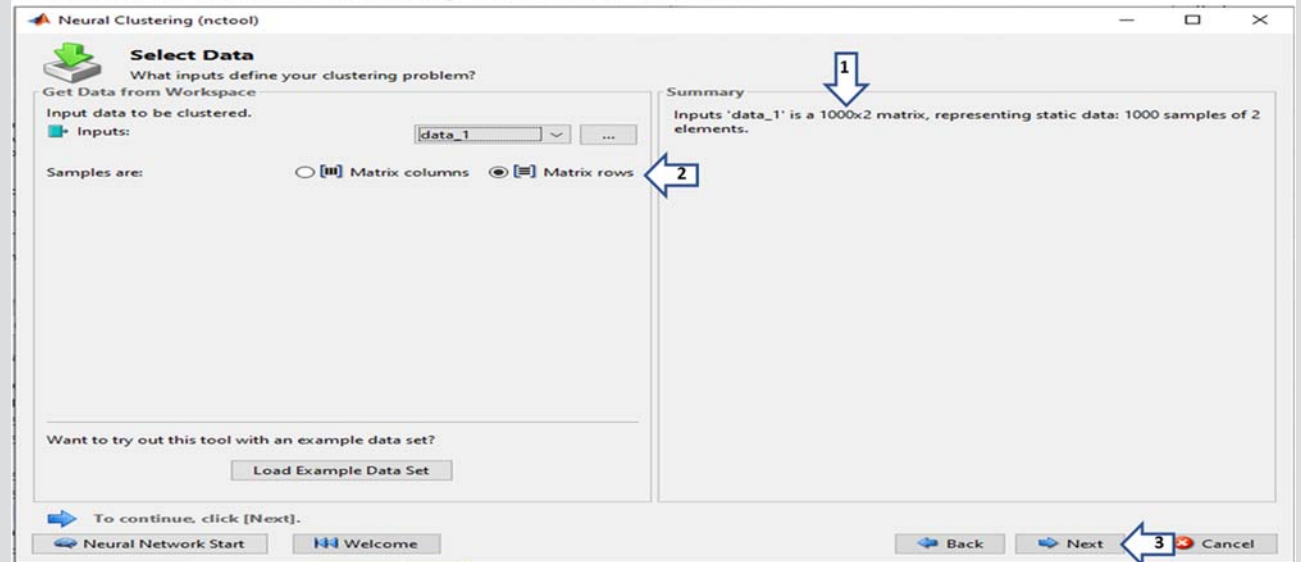
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

6

In the MATLAB "Neural Network Clustering" window screen: "Select Data"

In the "Summary" section verify that the "input data is 1000 x 2 matrix," specify the target as "Matrix rows," and click in the button "Next"

6. In the MATLAB "Neural Network Clustering" window screen: "Select Data"



In the "Summary" section verify that the "input data is 1000 x 2 matrix", specify the target as "Matrix rows", and click in the button "Next".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 7
Description
In the MATLAB "Neural Network Clustering" window screen: "Network Architecture"
Define a "Self-Organizing Map (SOM)" with Input of two variables, a "SOM of 100 neurons in two-dimension matrix form of 10×10 " and "Output with 100 variables or classes." Then click in the "Next" button

Screen figure

7. In the MATLAB "Neural Network Clustering" window screen: "Network Architecture"

Neural Clustering (nctool)

Network Architecture

Set the number of neurons in the self-organizing map network.

Self-Organizing Map

Define a self-organizing map. (selforgmap)

Size of two-dimensional Map:

Recommendation

Return to this panel and change the number of neurons if the network does not perform well after training.

Restore Defaults

Neural Network

Input: 2

SOM Layer: W, +, 10 x 10

Output: 100

Change settings if desired, then click [Next] to continue.

Neural Network Start Welcome Next Cancel

Define a "Self-Organizing Map (SOM)" with Input of 2 variables, a "SOM of 100 neurons in two-dimension matrix form of 10×10 " and "Output with 100 variables or classes". Then click in the "Next" button.

8

In the MATLAB “Neural Network Clustering” window screen: “Train Network”

In the “Train Network” the “SOM algorithm” is used, the training stops automatically stop when the full number of epochs have occurred, by default they are 200. Click on the buttons: “SOM Topology” and “SOM Sample Hits”

8. In the MATLAB “Neural Network Clustering” window screen: “Train Network”

The screenshot displays the MATLAB Neural Network Clustering interface. The main window is titled "Neural Clustering (nctool)" and contains a "Train Network" section. Below this, there are instructions: "Train the network to learn the topology and distribution of the input samples." and "Train using batch SOM algorithm. (trainbu) (learnsomb)". A "Retrain" button is visible, with a blue arrow pointing to it labeled "1". To the right, a "Results" section contains buttons for "Plot SOM Neighbor Distances", "Plot SOM Weight Planes", "Plot SOM Sample Hits", and "Plot SOM Weight Positions". Below the instructions, a note states: "Training automatically stops when the full number of epochs have occurred." and another note says: "Training multiple times will generate different results due to different initial conditions and sampling." At the bottom of the window, there are buttons for "Neural Network Start", "Welcome", "Back", "Next", and "Cancel".

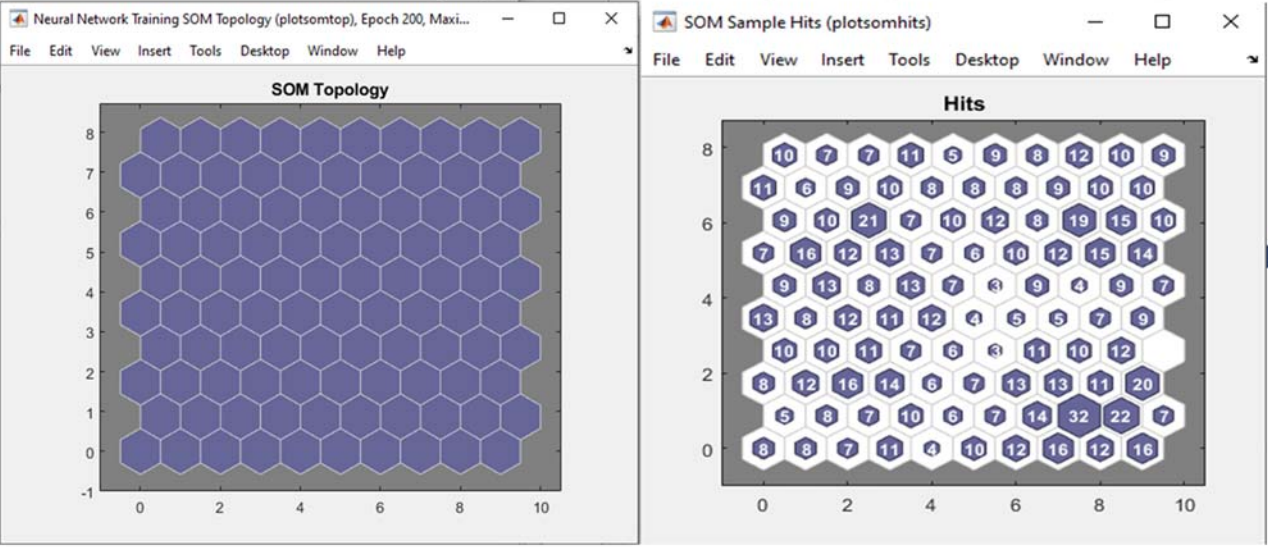
On the right side, a smaller window titled "Neural Network Training (nnrntool)" is open. It shows a diagram of the neural network with "Input" (2), "Layer" (W, +, 100), and "Output" (100). Below the diagram, the "Algorithms" section lists: "Training: Batch Weight/Bias Rules (trainbu)", "Performance: Mean Squared Error (mse)", and "Calculations: MATLAB". The "Progress" section shows a progress bar for "Epoch: 0" to "200" with "200 iterations" and "Time: 0.00.01". A blue arrow points to the "200" value labeled "2". The "Plots" section lists: "SOM Topology (plotsomtop)", "SOM Neighbor Connections (plotsomnc)", "SOM Neighbor Distances (plotsomnd)", "SOM Input Planes (plotsomplanes)", "SOM Sample Hits (plotsomhits)", and "SOM Weight Positions (plotsompos)". A blue arrow points to "SOM Topology" labeled "3" and another points to "SOM Sample Hits" labeled "4". The "Plot Interval" is set to "1 epochs". At the bottom, a green checkmark indicates "Maximum epoch reached." and there are "Stop Training" and "Cancel" buttons.

In the “Train Network” the “SOM algorithm” is used, the training stops automatically stop when the full number of epochs have occurred, by default they are 200. Click on the buttons: “SOM Topology” and “SOM Sample Hits”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide	Description	Screen figure
9	<p>In the MATLAB <i>“Neural Clustering”</i> window screen: <i>“Neural Network Training ‘SOM Sample Hits’ and ‘Plot SOM Weight Planes’</i></p> <p>The <i>“SOM Topology”</i> shows its default hexagonal topology 10×10 for the 100 neurons. The <i>“SOM Sample Hits,”</i> showing the 100 neurons locations and indicates their <i>“clusters centers.”</i> Where the <i>“maximum numbers of hits associated with any neurons is 32 input vectors in that cluster</i></p>	<p>9. In the MATLAB <i>“Neural Clustering”</i> window screen: <i>“Neural Network Training ‘SOM Sample Hits’ and ‘Plot SOM Weight Planes’</i></p>  <p>The <i>“SOM Topology”</i> shows its default hexagonal topology 10×10 for the 100 neurons . The <i>“SOM Sample Hits,”</i> showing the 100 neurons locations and indicates their <i>“clusters centers”</i> . Where the <i>“maximum numbers of hits associated with any neurons is 32 input vectors in that cluster.</i></p> <p>From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

10

In the MATLAB “Neural Clustering” window screen: “Neural Network Training,” Plot “SOM Input Planes” and “SOM ND”
Click on the buttons: “SOM Neighbor Distance” and “SOM Input Planes” then, click the “Next Button”

10. In the MATLAB “Neural Clustering” window screen: “Neural Network Training”, Plot “SOM Input Planes” and “SOM ND”

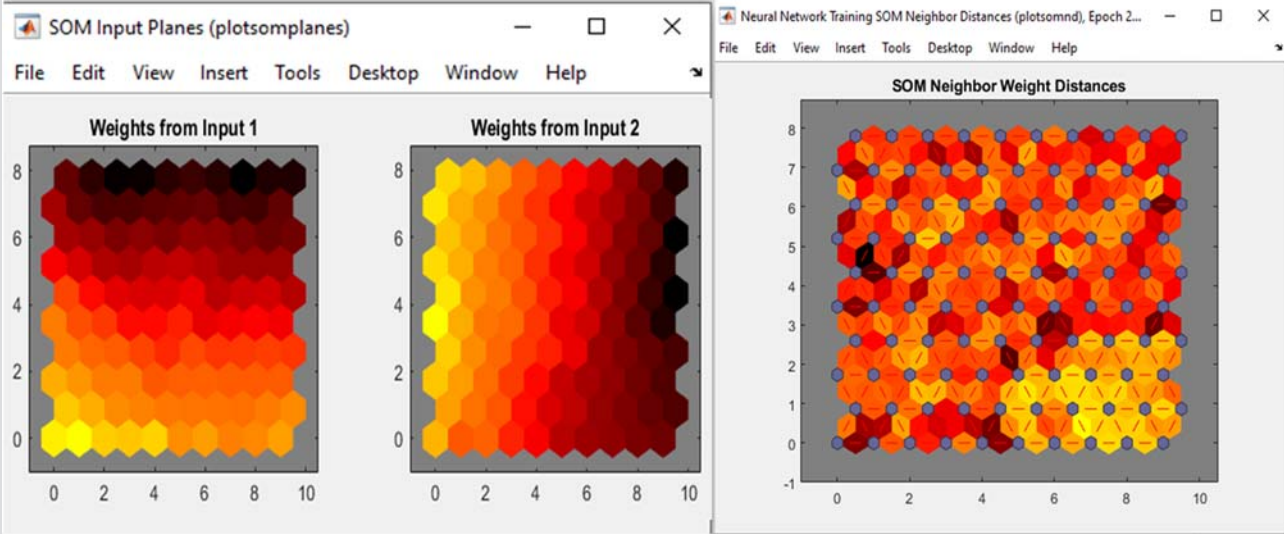
The screenshot displays two MATLAB windows. The left window, titled "Neural Clustering (nctool)", shows the "Train Network" interface. It includes a "Retrain" button and a "Next" button with a blue arrow pointing to it. The right window, titled "Neural Network Training (nntraintool)", shows a neural network diagram with 2 input nodes, 100 hidden nodes, and 100 output nodes. Below the diagram, there are sections for "Algorithms", "Progress", and "Plots". The "Plots" section lists several plots to be generated, including "SOM Neighbor Distances" and "SOM Input Planes", which are highlighted with blue arrows and numbers 1 and 2 respectively. A "Next" button is also present in this window.

Click on the buttons: “SOM Neighbor Distance” and “SOM Input Planes” then, click the “Next Button”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

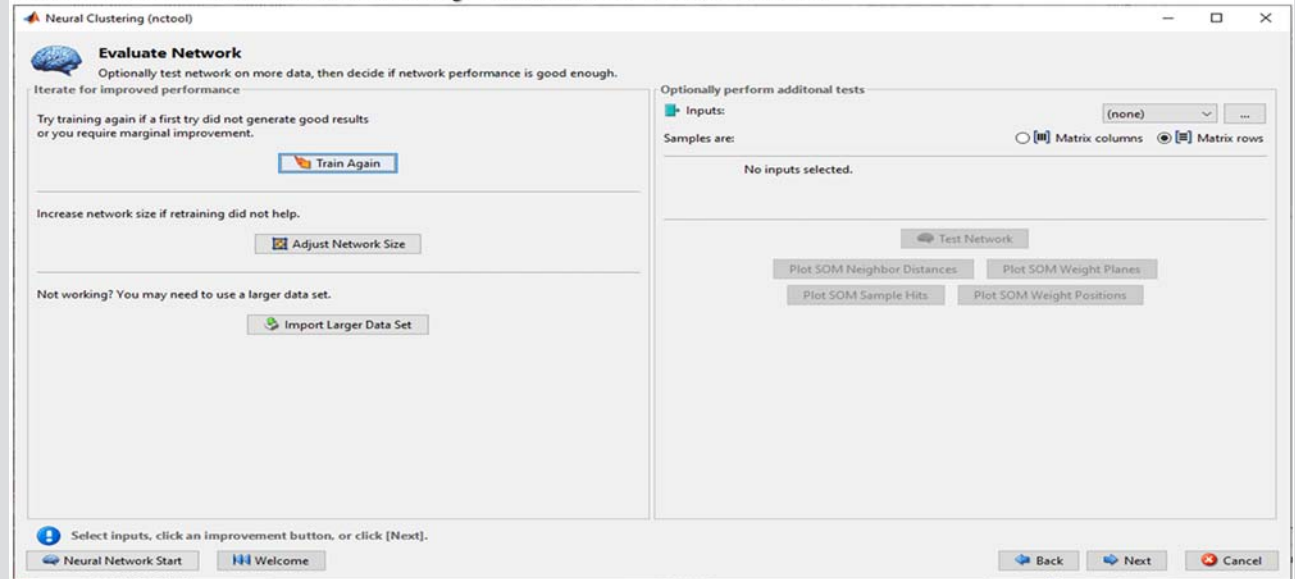
(Continued)

Slide	Description	Screen figure
11	<p>In the MATLAB "Neural Pattern Recognition" window screen: "Network Architecture"</p> <p>In "Plot SOM Input Planes" shows the visualization of the 2 inputs vector as "Weight Planes," where darker colors represent larger weights, for this dataset there is not correlations between the two variables. In the plot "SOM Neighbor Distances," the red lines show the distances between them and the connected neighboring neurons, the darker colors represent larger distances and the lighter colors smaller distances</p>	<p>11. In the MATLAB "Neural Clustering" window screen: "Neural Network Training", Plot "SOM Input Planes" and "SOM ND"</p>  <p>In "Plot SOM Input Planes" shows the visualization of the 2 inputs vector as "Weight Planes", where darker colors represent larger weights, for this data set there is not correlations between the 2 variables. In the plot "SOM Neighbor Distances", the red lines show the distances between them and the connected neighboring neurons, the darker colors represent larger distances and the lighter colors smaller distances.</p> <p>From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

In the MATLAB “Neural Network Clustering” window screen: “Evaluate Network”

Feel free to reevaluate the network to try to find better values with: “Train Again,” “Adjust Network Size,” “Import Larger Dataset,” “Optionally perform additional test,” etc. Then, click the “Next” button

12. In the MATLAB “Neural Network Clustering” window screen: “Evaluate Network”



Feel free to re-evaluate the network to try to find better values with: “Train Again”, “Adjust Network Size”, “Import Larger Data Set”, “Optionally perform additional test”, etc. Then, click the “Next” Button.

(Continued)

Slide 13 Description In the MATLAB “Neural Network Clustering” window screen: “Deploy Solution” Select the option needed for your “Deploy Solution”: In the section “Application Deployment” click button “MATLAB function”, In “Code Generation” click button “MATLAB Matrix-Only Function”, in “Simulink Deployment” click “Simulink Diagram” and in “Graphics” click button “Neural Network Diagram”. Click “Next” Button.

Screen figure

13. In the MATLAB “Neural Network Clustering” window screen: “Deploy Solution”

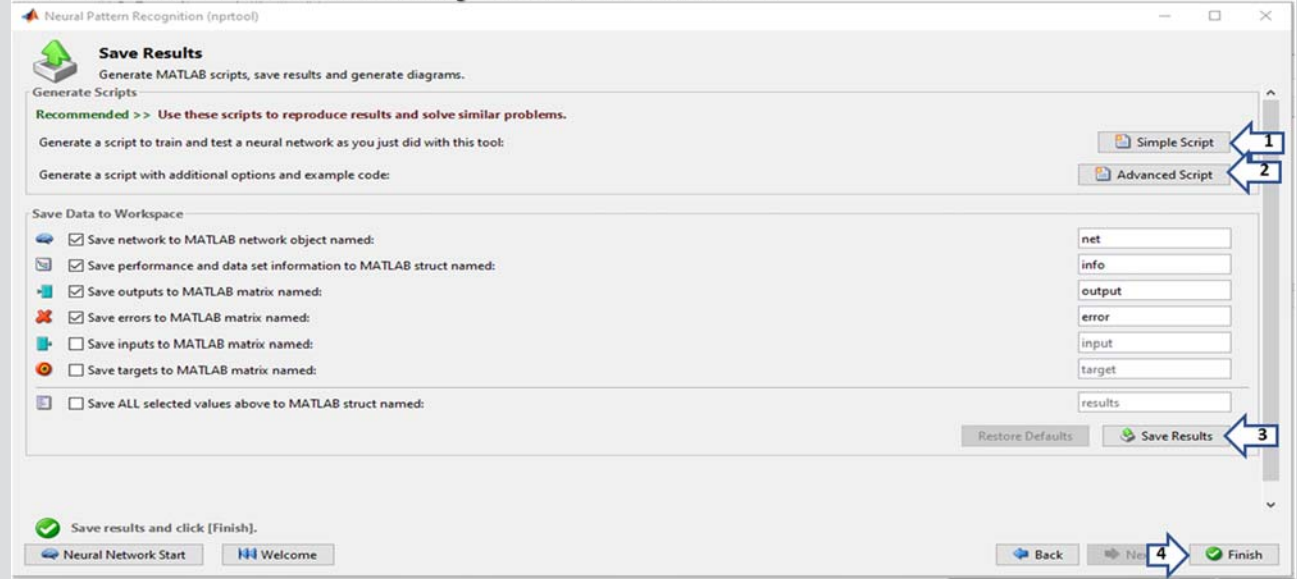
Select the option needed for your “Deploy Solution”: In the section “Application Deployment” click button “MATLAB function”, In “Code Generation” click button “MATLAB Matrix-Only Function”, in “Simulink Deployment” click “Simulink Diagram” and in “Graphics” click button “Neural Network Diagram”. Click “Next” Button.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

In the MATLAB “Neural Network Clustering” window screen: “Save Results”

In the section “Generate Scripts” click the buttons: “Simple Script” and “Advanced Script”. In the section “Save Data to Workspace” click button “Save Results”. Finally click on the “Finish” button, and its confirmation dialog

14. In the MATLAB “Neural Network Clustering” window screen: “Save Results”



In the section “Generate Scripts” click the buttons: “Simple Script” and “Advanced Script”. In the section “Save Data to Workspace” click button “Save Results”. Finally click on the “Finish” button, and its confirmation dialog

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide

Description

Screen figure

15

In MATLAB workspace save the functions and script generated by “Neural Network Clustering”
Save the two functions generated by MATLAB “Neural Network Clustering” as:
“myNeuralNetworkFunction.m” and
“myNeuralNetworkFunctionMatrix.m”. Save also the two scripts as:
“myNeuralNetworkSimplyScript.m” and
“myNeuralNetworkAdvancedScript.m”. Note: They can be useful later for other projects

15. In MATLAB workspace save the functions and script generated by “Neural Network Clustering”

The screenshot displays the MATLAB environment. The workspace on the left shows several variables: colheaders (1x2 cell), data (1000x2 double), data_1 (1000x2 double), net1 (1x1 network), output (100x1000 double), output1 (100x1000 double), and textdata (1x2 cell). The current folder contains files like Visit_Measurements.csv, myNeuralNetworkSimplyScript.m, myNeuralNetworkFunctionMatrix.m, myNeuralNetworkFunction.m, myNeuralNetworkAdvancedScript.m, and others.

The editor window shows the function `myNeuralNetworkFunctionMatrix(x1)` and the `Self-Organizing Map (view)` diagram. The diagram illustrates a neural network with an input of 2, a layer of 100 neurons, and an output of 100. The diagram includes a weight matrix 'w', a summation node '+', and a transfer function block.

```
1 function [y1] = myNeuralNetworkFunctionMatrix(x1)
2 %MYNEURALNETWORKFUNCTION neural network simulation function.
3
4 % Auto-generated by MATLAB, 25-Mar-2020 12:11:29.
5
6 % [y1] = myNeuralNetworkFunction(x1) takes these arguments:
7 % x = Qx2 matrix, input #1
8 % and returns:
9 % y = Qx100 matrix, output #1
10 % where Q is the number of samples.
11
12 %#ok<<RPS10>
13
14 % ===== NEURAL NETWORK CONSTANTS =====
15
16 % Layer 1
17 IW1_1 = [0.94671666666666666508 0.8354166666666666658525;0.977149999999999996305 0.66554999999999996393;0.885549999999999996948
18
19 % ===== SIMULATION =====
20
```

Save the two functions generated by MATLAB “Neural Network Clustering” as: “myNeuralNetworkFunction.m” and
“myNeuralNetworkFunctionMatrix.m”. Save also the two scripts as: “myNeuralNetworkSimplyScript.m” and
“myNeuralNetworkAdvancedScript.m”. Note: They can be useful later for other projects.

16

Closing MATLAB tools and Plots generated by "Neural Network Clustering" application
Close without saving the "Simulink diagram for the ANN" and the "Self-Organizing Map generated" and the "Neural Network—Neural Clustering App"

16. Closing MATLAB tools and Plots generated by by "Neural Network Clustering" application

The screenshot displays the MATLAB Simulink environment. At the top, the title bar reads "untitled * - Simulink". The main workspace shows a Simulink diagram with a "Constant" block (x1) connected to an "Input NNET" block, which is connected to an "Output" block (y1). The "Input NNET" block is labeled "Self-Organizing Map".

Below the main workspace, there are two smaller windows:

- Neural Network Start (nnstart):** A window titled "Welcome to Neural Network Start" with a sub-header "Learn how to solve problems with neural networks." It contains a "Getting Started Wizards" tab and a "More Information" tab. Under "More Information", there are four links: "Getting started documentation. [Neural Network Guide](#)", "Neural network demonstrations. [List of Examples](#)", "Neural network datasets. [List of Datasets](#)", and "Neural network textbook demonstrations. [List of Textbook Examples](#)".
- Self-Organizing Map (view):** A window showing a detailed view of a neural network layer. It has an "Input" block with a value of 2, a "Layer" block containing a weight matrix "W", an addition node "+", and a plot icon, and an "Output" block with a value of 100.

At the bottom of the Simulink interface, the status bar shows "Ready", "80%", and "VariableStepAuto".

Close without saving the "Simulink diagram for the ANN" and the "Self-Organizing Map generated" and the "Neural Network – Neural Clustering App".

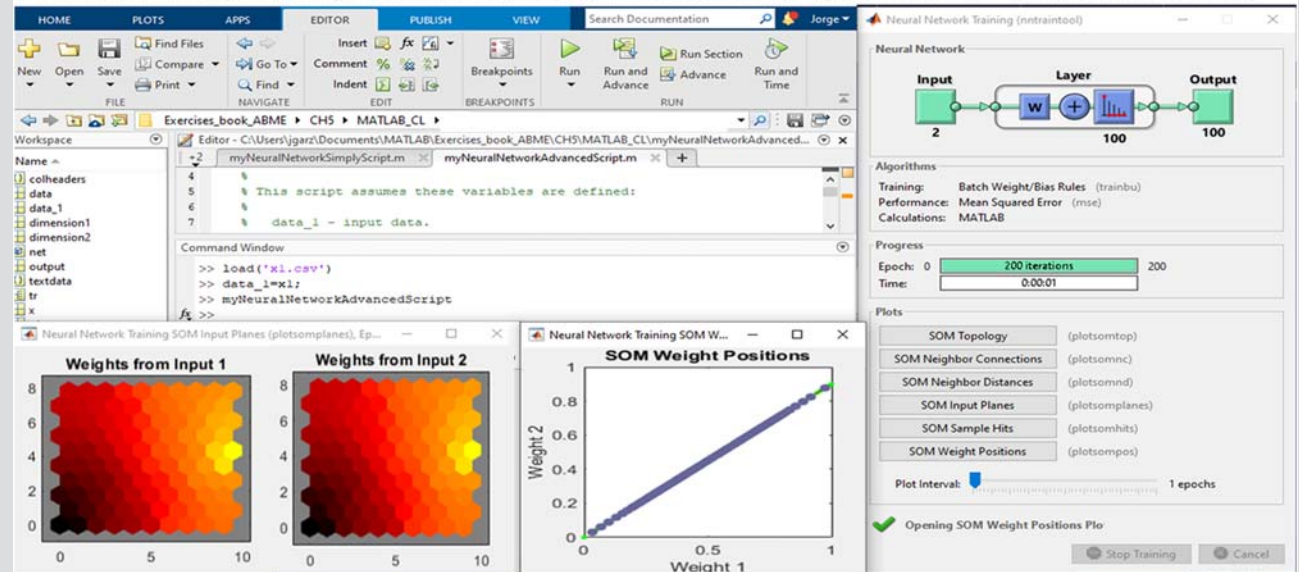
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 17 Description In MATLAB workspace testing new data in "myNeuralNetworkAdvancedScript" generated for the "Neural Pattern Clustering". To evaluate a new data as a matrix, load the file "X1.csv," make the variable "data_1 = x1," and run the "advance script" as shown. The "Neural Network Training screen" will train the new data, after 200 iterations it stop the training. Click on "SOM Neighbor Distances" button and in the "SOM Weight Position." The results show that there are "positive correlations between the two signals." Finally, close all

Slide 17 Screen figure 17. In MATLAB workspace testing new data in "myNeuralNetworkAdvancedScript" generated for the "Neural Pattern Clustering"



To evaluate a new data as a matrix, load the file "X1.csv," make the variable "data_1=x1," and run the "advance script" as shown. The "Neural Network Training screen" will train the new data, after 200 iterations it stop the training. Click on "SOM Neighbor Distances" button and in the "SOM Weight Position". The results show that are "positive correlations between the two signals". Finally, close all.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

"An AI model *MATLAB Deep Learning Toolbox* was obtained using a *Neural Network for clustering* based on *Self-Organizing MAP* through a *Shallow Neural Network*, with the purpose of visually analyzing a dataset," that contains a matrix based on two vector normalized sEMG signals from "type 2 DM patients' right limb soleus muscle" in two consecutive monthly visits. "No correlation between the two signals were found, indicating an advance on the Soleus muscle affectionation response by the type 2 DM." Finally, using the "*Neural Network Advanced MATLAB Script*" generated for the "*Neural Pattern Clustering*" a matrix based on two vector inputs from two consecutive datasets of "healthy patient right limb soleus muscle" obtained by "sEMG signals" on two consecutive monthly visits shows a positive correlations between them, indicating no significative changes to the behavior of the healthy "soleus muscle."

Recommendation

This kind of "*Self-Organizing MAP*, also known as *Korhonen Network (KN)*, through a *Shallow Neural Network*" can be applied to visually analyze biomedical signals from different bioinstruments.

5.3.3 Research 5.3 Neural Network for Dynamic Time series based on a "NARX is a nonlinear autoregressive exogenous model" to analyze vertical Ground Reaction Forces signals

5.3.3.1 Case for research

Obtain an AI graphic representation model using *MATLAB Deep Learning Toolbox* using a *Neural Network for Dynamic Time Series* based on a "*nonlinear autoregressive network with exogenous inputs (NARX) model*" to visually analyze the difference between "normalized average vGRF" signals.

5.3.3.2 General objective

Apply "*MATLAB Deep Learning Toolbox*" to define, build, train, and deploy a "*Nonlinear autoregressive with External (Exogenous) Input (NARX)*" to visually analyze and differentiate between two normalized average vGRF signals from the right limb during "*brisk walking*": one of "*healthy people*" used as a reference versus "*unhealthy patients*."

5.3.3.3 Specific objectives

- Load as an input to the "NARX" the "vGRF_right_strides_patients.csv" dataset file that contains one

vector of the "normalized average vGRF" obtained from the right limb during 3 minutes of "*brisk walking*" of an "*unhealthy patient*."

- Note: "*brisk walking*" is means the walking faster than you would normally, typically it is around 100 steps per minutes. "*Brisk walking*" is considered a moderate intensity walking.
- Load as a target to the "NARX" the "vGRF_right_strides_ref.csv" dataset file that contains one vector of the "normalized average vGRF" obtained from the right limb during 3 minutes of "*brisk walking*" of 10 "*healthy people*."
- Randomly divide the datasets in optimal percentages numbers in parts for: "*Training*," "*Validation*," and "*Testing*" of the "NARX."
- Select the optimal number of "*Hidden Neurons layer*" and number of delays for the best performance in the "NARX."
- Train the dataset using a "*Bayes Regularization algorithm*" for the "NARX."
- Visualize the "NARX" behaviors using plots for:
 - "*Output element for the time-series for the healthy people*" used as a reference versus "*unhealthy patient*."
 - "*Error histogram*" obtain from "*time-series response between the two signals: reference versus unhealthy patient*."
 - "*Error Input-Error Cross-correlation*" that indicates the differences between the "*Target-Output*."
 - "*Error Input-Error Cross-correlation*" that indicates the differences between the "*Target-Output*."
- Deploy the MATLAB "*Neural Network Time series*" AI model as a MATLAB function and as a "*Neural Network Advanced Script*."
- Compare the results obtain using the "NARX" versus "*a normal time plot*" and make observations in the "*AI NARX Model*" obtained.

5.3.3.4 Background for "vertical Ground Reaction Forces"

In "*Biomechanics*," the "*Ground Reaction Force (GRF)*" is the force exerted by the ground on a body in contact with it. For example, a person standing motionless on the ground exerts a contact force on it that is equal to the person's weight, and at the same time an equal and opposite force is exerted by the ground on the person. We can measure the forces involved in the stance phase of the human gait; this is called "*kinetics*," which is the study of movement and the forces involved in producing it.

The force measured takes advantage of “*Newton’s third law of motion: for every action, there is an equal and opposite reaction.*” When we step on the ground, we produce a vector of force that is generally downward and backward. The ground produces a force that is generally upward, lateral, and forward, and it is the *Ground Reaction Force (GRF)* that is measured by a “*force plate.*” The GRF coincides with the notion of a normal force; this means the perpendicular component to the surface. In a more general case, the “*GRF*” will also have components in parallel to the ground, this is case while the person is walking making motions that require the frictional forces with the ground. The resulting force has three component dimensions (3D): “*vertical Ground Reaction forces (vGRF),*” “*medial-lateral Ground Reaction forces (mlGRF),*” and “*anterior-posterior Ground Reaction forces (apGRF)*” [19].

“*Pathologic human gaits attributed to neurological conditions with their related diseases and movement disorders*” and other illness can be detected with a synchronized set of “*bioinstruments*” to evaluate the body signals, such as the “*vertical Ground reaction Forces (vGRF)*” and other signals obtained from the human body, as explained

in my previous book: “*Applied Biomechatronics Using Mathematical Models*” [30].

5.3.3.5 Dataset

The dataset “*vGRF_right_strides_patient.csv*” contains “*normalized average vGRF*” values obtained from the right limb during 3 minutes of “*brisk walking*” of an “*unhealthy patient,*” as shown in Table 5.7, and the dataset “*vGRF_right_strides_ref.csv*” contains “*normalized average vGRF*” values obtained from the right limb during 3 minutes of “*brisk walking*” of an “*healthy patients.*” They are used as a reference to compare all patients as indicated in Table 5.8.

5.3.3.6 Procedure

The steps to obtain an AI model “*using a Neural Network for Dynamic Time series based on a Recurrent Neural Network-Nonlinear Autoregressive to analyze vertical Ground Reaction Forces signals*” “are summarized in Table of slides 5.3, and each step of the example is visually explained using screen sequences with instructions in easy to follow screen figures.

TABLE 5.7 Dataset “vGRF_right_strides_patients.csv” fields and descriptions.

Field	Description * 111 instances. Number of fields = 1
vGRFr	Average of Vertical Ground Reaction Forces of right limb

Note: This dataset is available in the companion directory of the book, in the following directory: “... \Exercises_book_ABME\CH5\MATLAB_DTS\vGRF_right_strides_patient.csv.”

TABLE 5.8 Dataset “vGRF_right_strides_ref.csv” fields and descriptions.

Field	Description *111 instances. Number of fields = 1
vGRFr_ref	Average of Vertical Ground Reaction Forces of right limb of healthy people with normal gaits

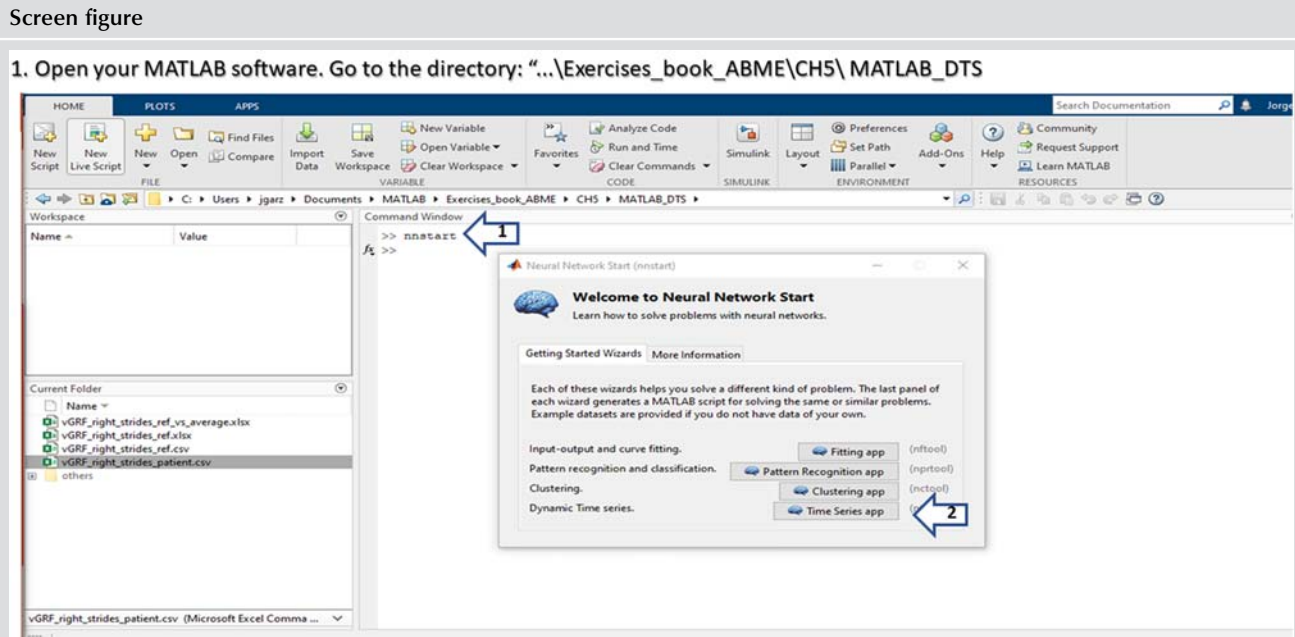
Note: This dataset is available in the companion directory of the book, in the following directory: “... \Exercises_book_ABME\CH5\ MATLAB_DTS\vGRF_right_strides_ref.csv.”

Table of slides 5.3 Steps for MATLAB Deep Learning Toolbox using a Neural Network for Dynamic Time series based on an “NARX model” to analyze vertical Ground Reaction Forces signals.

Slide 1 Description

Open your MATLAB software. Go to the directory: “... \Exercises_book_ABME\CH5\ MATLAB_DTS”

In the command prompt enter the command for Neural Network Starter: `nnstart`, “MATLAB Deep Learning Toolbox” includes a “Neural Network GUI—`nnstart`” that opens a window with launch buttons for “Neural Network Fitting,” “Pattern Recognition,” “Clustering,” and “Time series tools.” Click the button: “Dynamic Time Series (`ntstool`)”



In the command prompt enter the command for Neural Network : MATLAB Deep Learning Toolbox™ includes a “Neural Network GUI – `nnstart`” that opens a window with launch buttons for “Neural Network Fitting”, “Pattern Recognition”, “Clustering” and “Time series tools”. Click the button: “Dynamic Time Series (`ntstool`)”

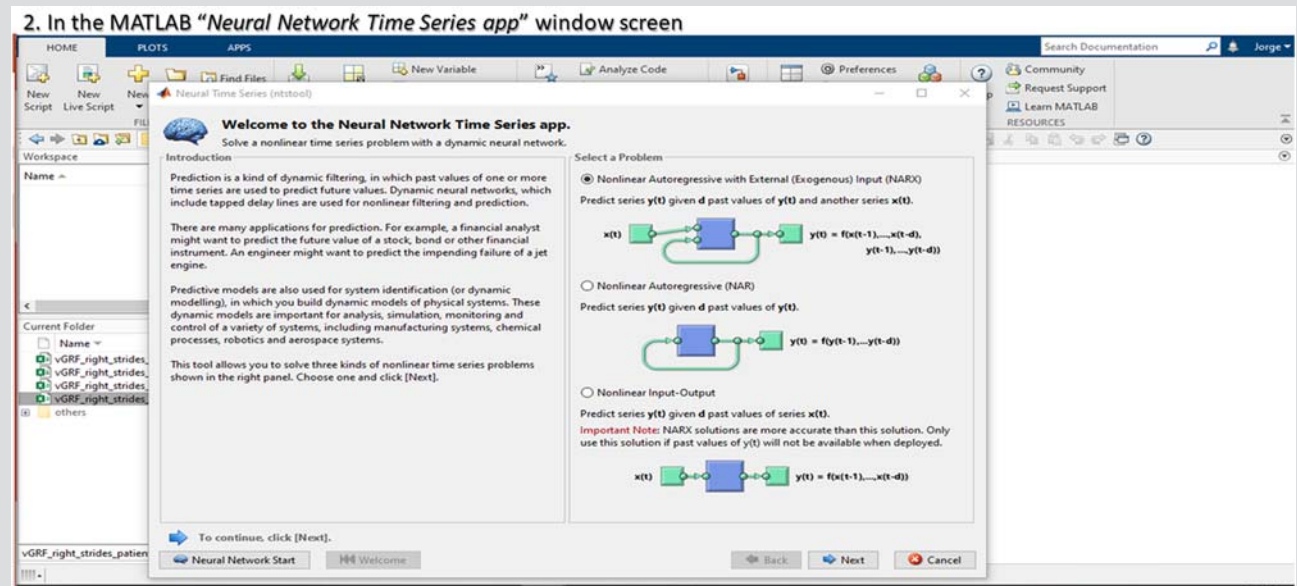
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description Screen figure

2 In the MATLAB "Neural Network Time Series app" window screen "Neural Network Time Series" is a tool that can be used as a filter values or predict futures values in time series based in past values. This app allows to resolve three dynamic time series problems: "Nonlinear Autoregressive with External Inputs (NARX)," "Nonlinear Autoregressive (NAR)" and "Nonlinear Input-Output." Select the first option "NARX," then click the button "Next"



"Neural Network Time Series" is a tool that can be used as a filter values or predict futures values in time series based in past values. This app allows to resolve three dynamic time series problems: "Nonlinear Autoregressive with External Inputs (NARX)," "Nonlinear Autoregressive (NAR)" and "Nonlinear Input-Output". Select the first option "NARX", then click the button "Next"

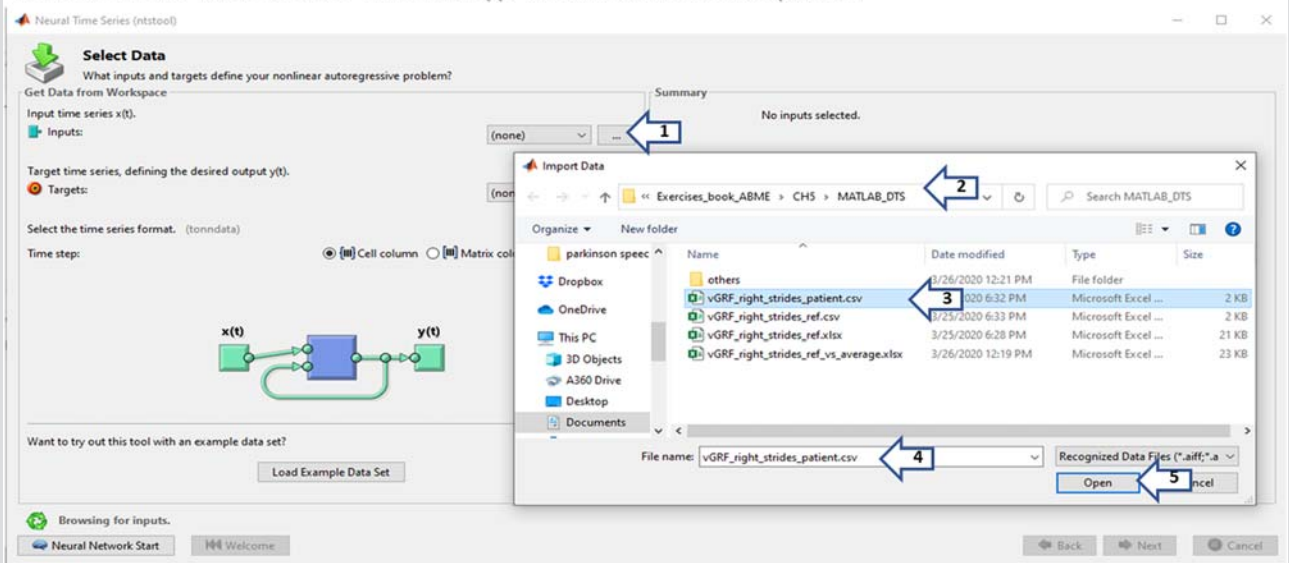
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

3

In the MATLAB "Neural Network Time Series app" window screen: "Select input Data"

Select the button for select the "input dataset ...," select the file "...\Exercises_book_ABME\CH5\MATLAB_DTS\lvGRF_right_strides_patient.csv," and click the button "Next"

3. In the MATLAB "Neural Network Time Series app" window screen: "Select input Data"



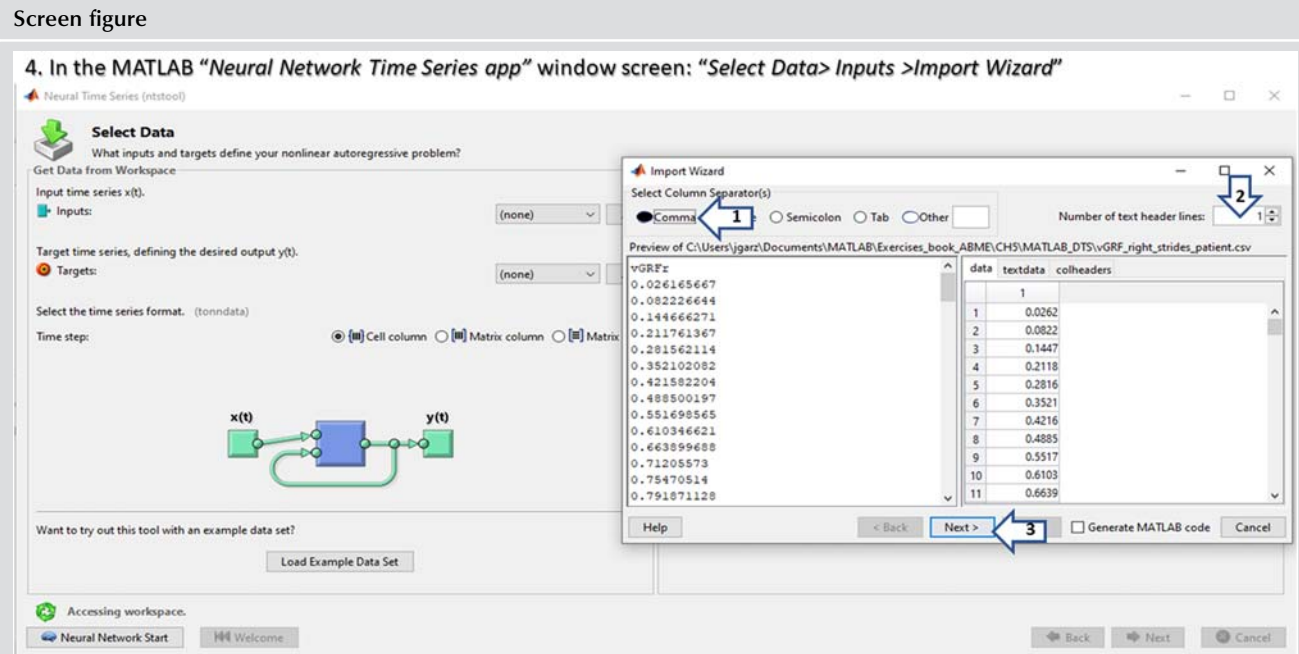
Select the button for select the "input dataset ...", select the file "...\Exercises_book_ABME\CH5\MATLAB_DTS\lvGRF_right_strides_patient.csv", and click the button "Next"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 4 Description
In the MATLAB "Neural Network Time Series app" window screen: "Select Data > Inputs > Import Wizard"
Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" button



Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" Button

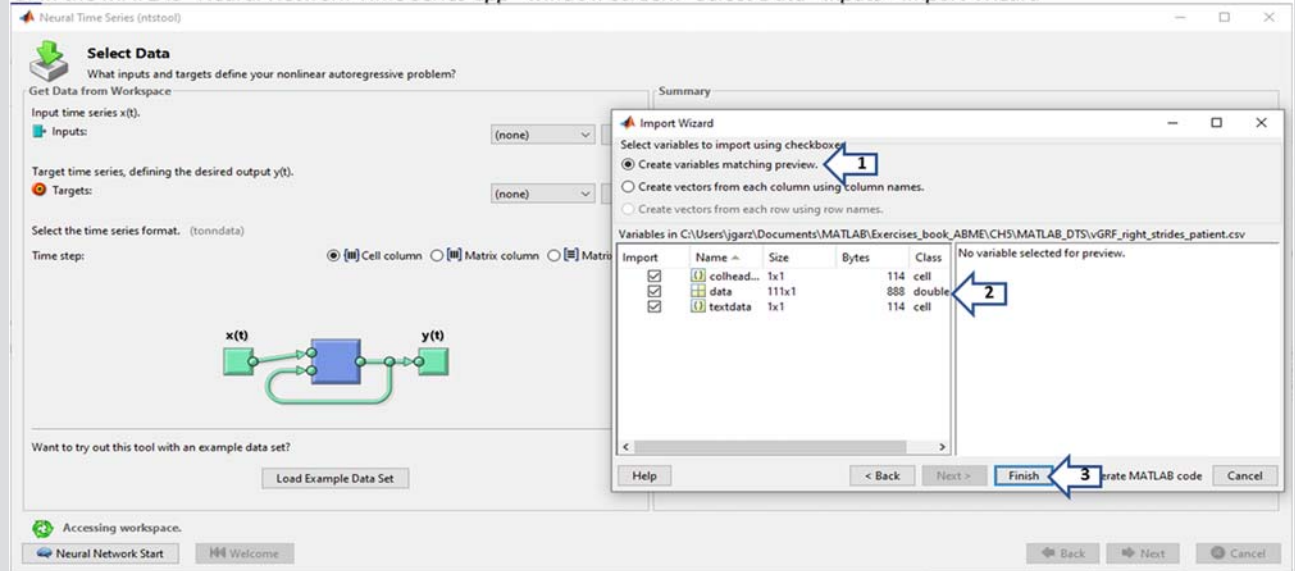
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

5

In the MATLAB “Neural Network Time Series app” window screen: “Select Data > Inputs > Import Wizard”

In the section “Select variable to import using checkboxes” activate “Create variables matching preview,” verify the creation of the three variables and click the “Finish” button

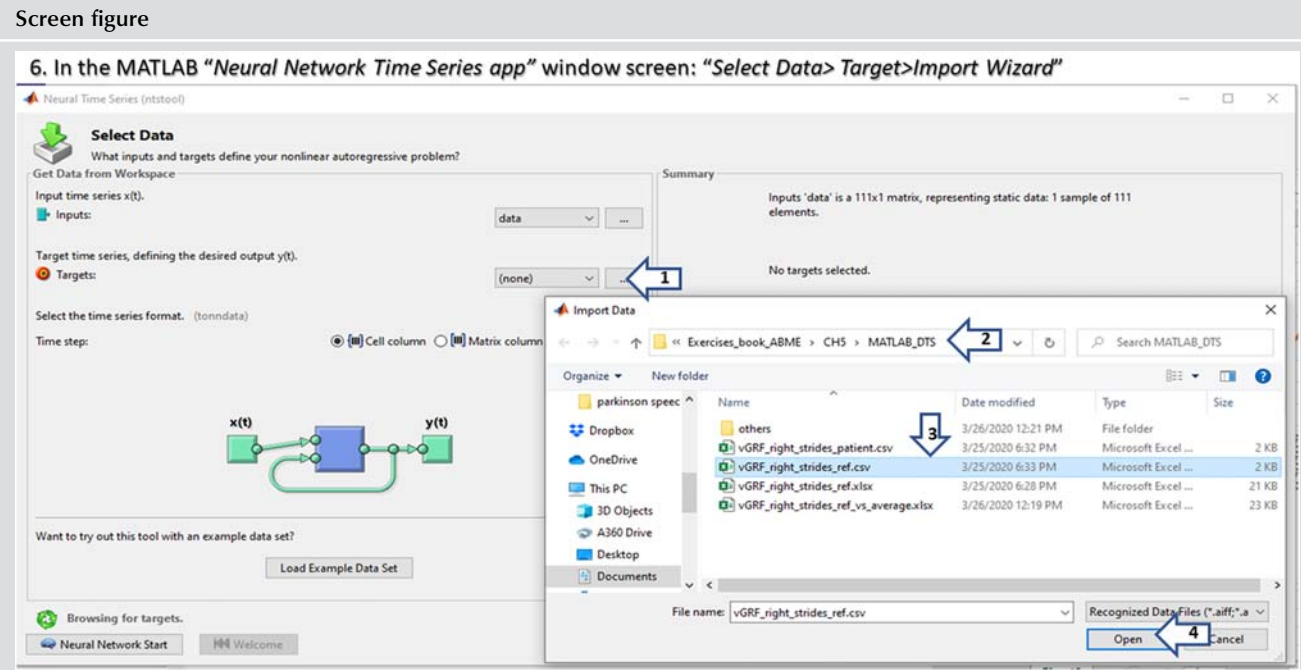
5. In the MATLAB “Neural Network Time Series app” window screen: “Select Data > Inputs > Import Wizard”



In the section “Select variable to import using checkboxes” activate “Create variables matching preview”, verify the creation of the 3 variables and click the “Finish” button.

(Continued)

Slide 6 Description
In the MATLAB "Neural Network Time Series app" window screen: "Select Data > Target > Import Wizard"
Select the button for select the "input dataset ...," select the file "... \Exercises_book_ABME\CH5 \MATLAB_DTS \vGRF_right_strides_ref.csv," and click the button "Open"



Select the button for select the "input dataset ...", select the file "... \Exercises_book_ABME\CH5\MATLAB_DTS\vGRF_right_strides_ref.csv", and click the button "Open".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

7

In the MATLAB “Neural Network Time Series app” window screen: “Select Data > Target > Import Wizard”
Select “Other” as the column separator, verify that the “Number of test header line is 1,” and click on the “Next” button

7. In the MATLAB “Neural Network Time Series app” window screen: “Select Data > Target > Import Wizard”

Summary: Inputs 'data' is a 111x1 matrix, representing static data: 1 sample of 111 elements.

Select Column Separator(s):
 Comma Space Semicolon Tab Other

Number of text header lines: 1

Preview of C:\Users\jgarza\Documents\MATLAB\Exercises_book_ABME\CH5\MATLAB_DTS\vGRF_right_strides_ref.csv

	data	textdata	colheaders
		1	
1	0.0312		
2	0.0932		
3	0.1626		
4	0.2374		
5	0.3153		
6	0.3942		
7	0.4719		
8	0.5466		
9	0.6171		
10	0.6823		
11	0.7418		

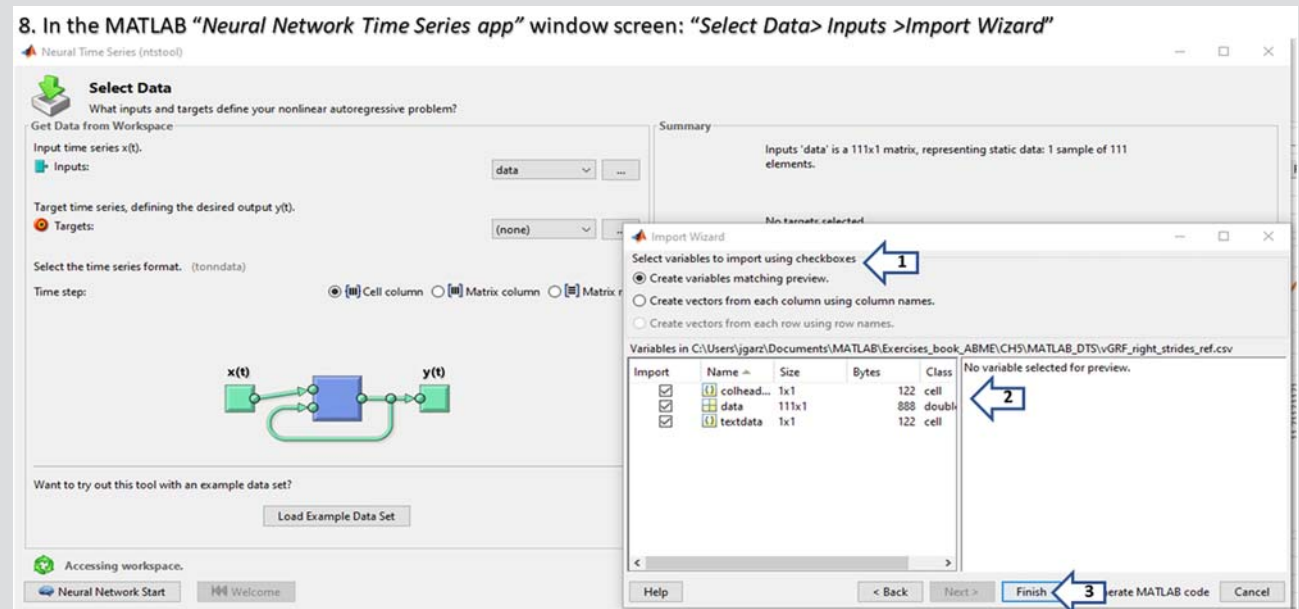
Select “Other” as the column separator, verify that the “Number of test header line is 1”, and click on the “Next” Button.

(Continued)

Slide Description

Screen figure

8 In the MATLAB "Neural Network Time Series app" window screen: "Select Data > Inputs > Import Wizard"
"Select variable to import using checkboxes" activate "Create variables matching preview," verify the creation of the three variables and click the "Finish" button



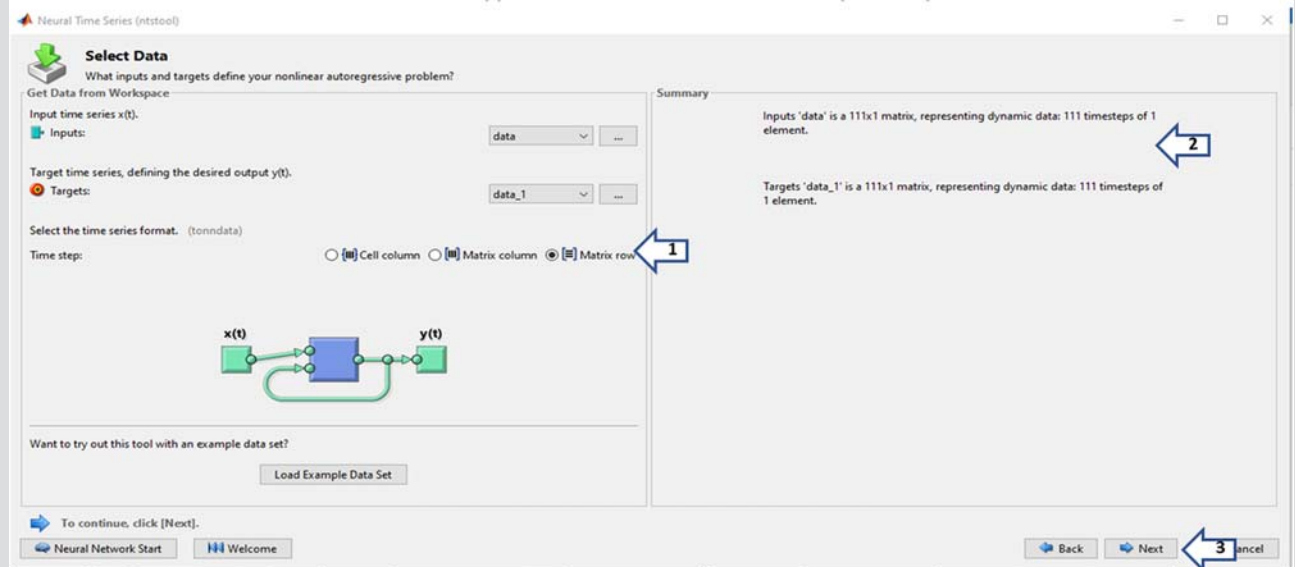
In the section "Select variable to import using checkboxes" activate "Create variables matching preview", verify the creation of the 3 variables and click the "Finish" button.

9

In the MATLAB “Neural Network Time Series app” window screen: “Select Data > Inputs > Import Wizard”

In the section “Select data” verify that the input and out dataset are specified, activate “Time steps” as “Matrix rows,” at the section “Summary” verify the input and outputs are 111-time steps \times 1 element. Finally, click the “Next” button

9. In the MATLAB “Neural Network Time Series app” window screen: “Select Data > Inputs > Import Wizard”

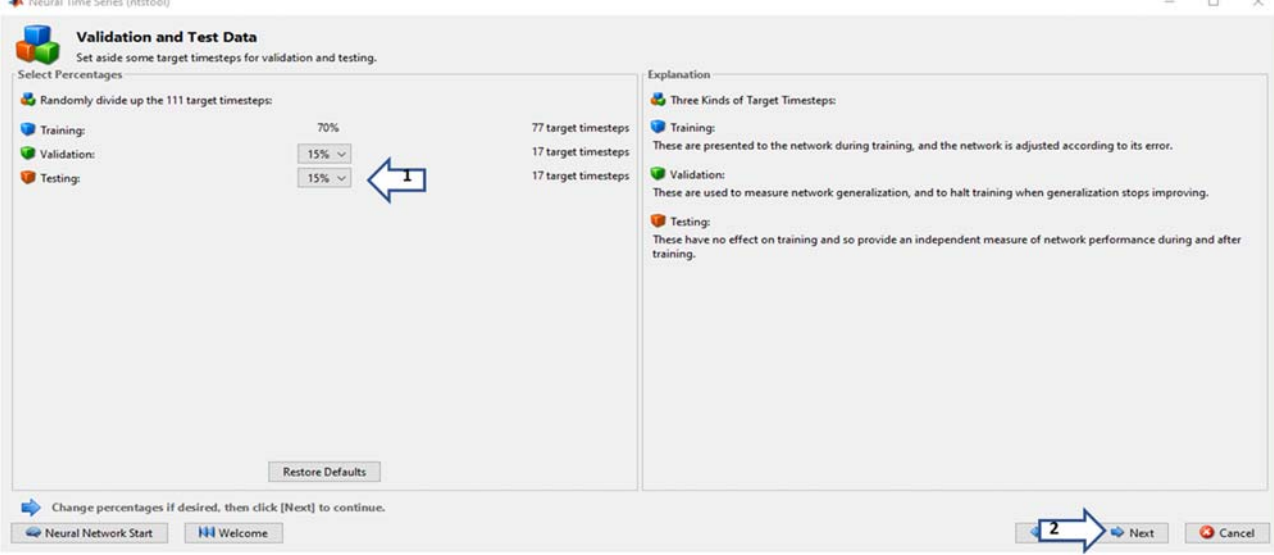


In the section “Select data” verify that the input and out dataset are specified, activate “Time steps” as “Matrix rows”, at the section “Summary” verify the input and outputs are 111 time steps x 1 element. Finally, click the “Next” button.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide	Description	Screen figure
10	In the MATLAB "Neural Network Time Series app" window screen: "Validation and test data" In the section "Select Percentages" verify that: "Training = 70%," "Validation = 15%," and "Testing = 15%," then click in the button "Next"	<p data-bbox="673 276 1616 308">10. In the MATLAB "Neural Network Time Series app" window screen: "Validation and test data"</p>  <p data-bbox="683 876 1906 933">In the section "Select Percentages" verify that: "Training=70%," "Validation=15%" and "Testing=15%," then click in the button "Next".</p> <p data-bbox="942 958 1792 990">From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

11

In the MATLAB "Neural Network Time Series app" window screen: "Network Architecture"

In the "Architecture Choices" the neural network is already defined as "nonlinear autoregressive network with exogenous inputs (NRAX)," for the "number of Hidden Neurons specify 15," in the "Number of delays 'd' specify 3." Note: These numbers could be adjusted later to improve ANN-NARX performance. Then click in the button "Next"

11. In the MATLAB "Neural Network Time Series app" window screen: "Network Architecture"

The screenshot shows the MATLAB "Neural Network Time Series app" window in the "Network Architecture" section. The "Architecture Choices" panel includes the following settings:

- Define a NARX neural network: (nars.net)
- Number of Hidden Neurons: 15
- Number of delays d: 3
- Problem definition: $y(t) = f(x(t-1), \dots, x(t-d), y(t-1), \dots, y(t-d))$

A "Recommendation" box states: "Return to this panel and change the number of neurons or delays if the network does not perform well after training. network will be created and trained in open loop form as shown below. Open loop (single-step) is more efficient than closed loop (multi-step) training. Open loop allows us to supply the network with correct past outputs as we train it to produce the correct current outputs. After training, the network may be converted to closed loop form, or any other form, that the application requires."

The "Neural Network" diagram shows an input layer with $x(t)$ and $y(t)$ (both with a bias of 1), a hidden layer with 15 neurons, and an output layer with 1 neuron. The hidden layer is labeled "Hidden Layer with Delays" and includes weights W and bias b . The output layer also includes weights W and bias b .

At the bottom, there are buttons for "Neural Network Start", "Welcome", "Next", and "Cancel". The "Next" button is highlighted with a blue arrow and the number 3.

In the "Architecture Choices" the neural network is already defined as "nonlinear autoregressive network with exogenous inputs (NRAX)", for the "number of Hidden Neurons specify 15", in the "Number of delays 'd' specify 3". Note: These numbers could be adjusted later to improve ANN-NARX performance. Then click in the button "Next".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

12 In the MATLAB "Neural Network Time Series app" window screen: "Train Network"
"Train Network"
In the "Train Network" section choose: "Bayesian Regularization," this algorithm takes more time, but usually give better results for some dataset and training stop when a minimum is reached on "Regularization (R) or Means Squared Error (MSE)." Click on "Train/Retrain" button until obtain lowers values of "R" and "MSE." Check the "Progress" section to observe the result, then in the "Plots" section click "Performance" button and observe "Best Validation Performance as 2.9933e⁻¹⁰ at epoch 1000"

12. In the MATLAB "Neural Network Time Series app" window screen: "Train Network"

The screenshot displays the MATLAB Neural Network Time Series app interface. The main window is titled "Neural Time Series (ntstool)". The "Train Network" section is active, showing a dropdown menu set to "Bayesian Regularization". The "Results" table shows the following values:

Target Values	MSE	R
Training: 77	2.99331e-10	9.99999e-1
Validation: 17	0.00000e-0	0.00000e-0
Testing: 17	1.38753e-8	9.99999e-1

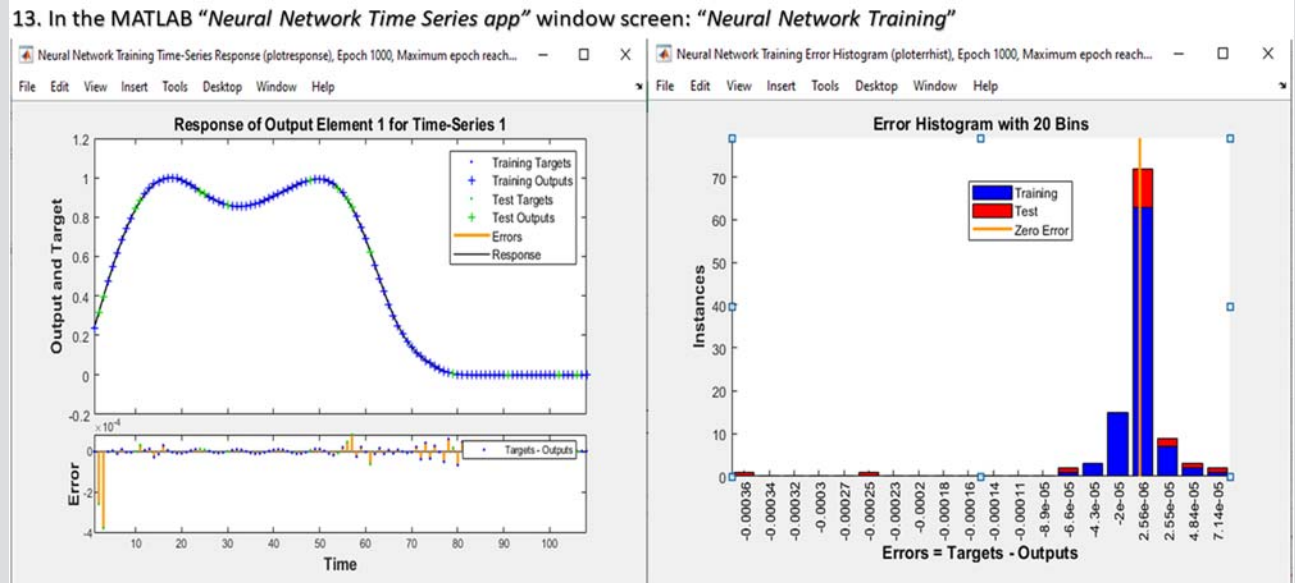
The "Progress" section shows a progress bar for 1000 iterations, with the current epoch at 1000. The "Plots" section has a "Performance" button highlighted with a blue arrow. A "Neural Network Training Performance" plot is open, showing Mean Squared Error (mse) on a logarithmic scale vs Epochs. The plot title is "Best Training Performance is 2.9933e-10 at epoch 1000". The plot shows three lines: Train (blue), Test (red), and Best (green). The Best line reaches its minimum at epoch 1000.

In the "Train Network" section choose: "Bayesian Regularization", this algorithm takes more time, but usually give better results for some dataset and training stop when a minimum is reached on "Regularization (R) or Means Squared Error (MSE)". Click on "Train/Retrain" button until obtain lowers values of "R" and "MSE". Check the "Progress" section to observe the result, then in the "Plots" section click "Performance" Button, and observe "Best Validation Performance as 2.9933e⁻¹⁰ at epoch 1000".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

13

In the MATLAB "Neural Network Time Series app" window screen: "Neural Network Training"
Click on the button: "Time-Series Response" to see the left plot of "Output and Target versus Time" and the "Error versus Time," that indicates the behavior of: "Training Targets/Outputs", "Test Targets/Outputs," "Errors and response," and click the button "Error Histogram." Click the button "Error Histogram" to see the right plot for "Error Histogram with 20 Bins," where indicates that the training and test error are small



Click on the button: "Time-Series Response" to see the left plot of "Output and Target vs Time" and the "Error vs Time", that indicates the behavior of: "Training Targets/Outputs", "Test Targets/Outputs", "Errors and response", and click the button "Error Histogram" . Click the button "Error Histogram" to see the right plot for "Error Histogram with 20 Bins", where indicates that the training and test error are small.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

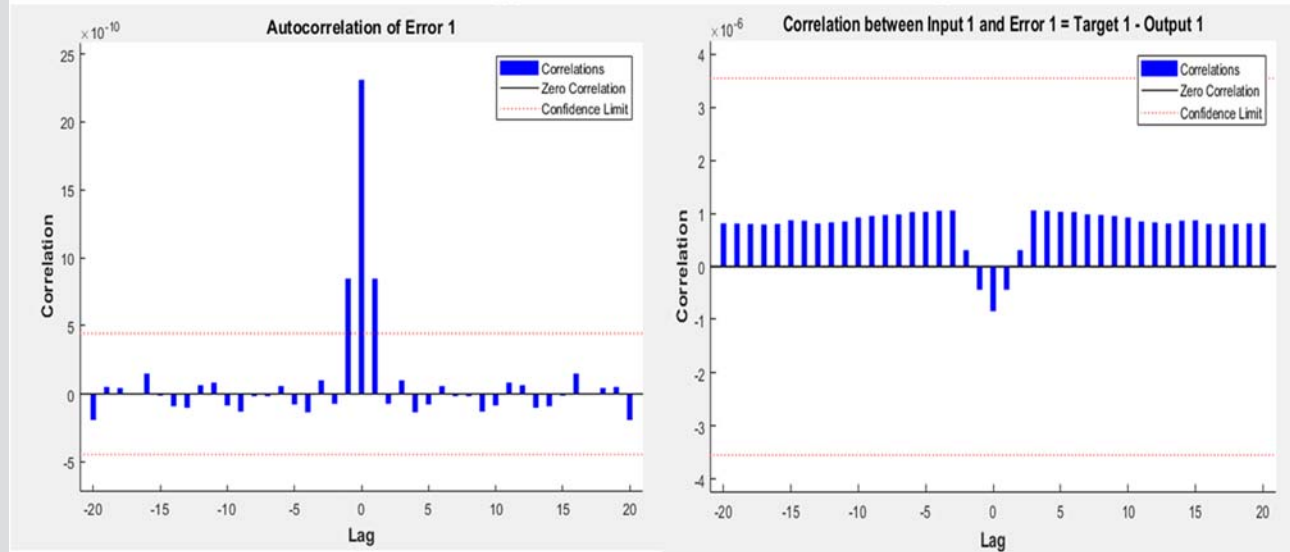
(Continued)

(Continued)

Slide 14 Description In the MATLAB "Neural Network Time Series app" window screen: "Neural Network Training" Click on the button: "Error Autocorrelation" to see the left plot of "Autocorrelation of error 1," and click the button "Error Input-Error Cross-correlation" that indicates the differences between the "Target and Output," showing lower values in the output obtained from the time series vector of the "vGRF_right_strides_patients.csv" with respect to the time series reference vector of the "vGRF_right_strides_ref.csv"

Screen figure

14. In the MATLAB "Neural Network Time Series app" window screen: "Neural Network Training"

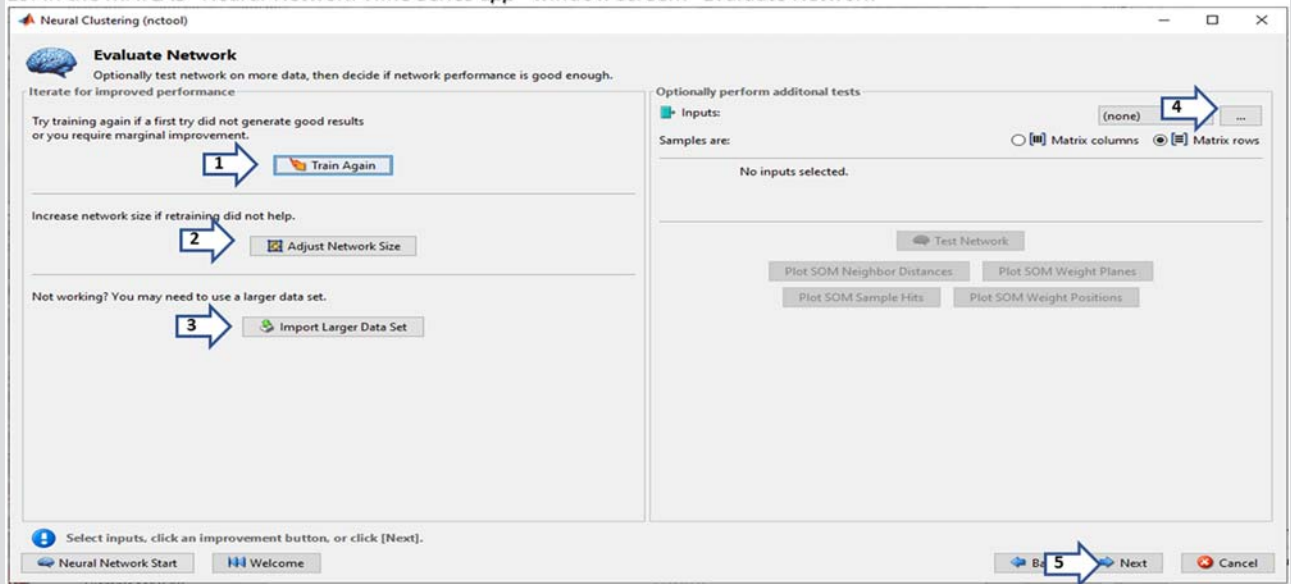


Click on the button: "Error Autocorrelation" to see the left plot of "Autocorrelation of error 1", and click the button "Error Input-Error Cross-correlation" that indicates the differences between the "Target - Output", showing lower values in the output obtained from the time series vector of the "vGRF_right_strides_patients.csv" with respect to the time series reference vector of the "vGRF_right_strides_ref.csv"

15

In the MATLAB "Neural Network Time Series app" window screen: "Evaluate Network"
Click in the "Train network" the "Next" button. Feel free to reevaluate the network to try to find better values with: "Train Again," "Adjust Network Size," "Import Larger Dataset," "Optionally perform additional tests," etc. Then, click the "Next button"

15. In the MATLAB "Neural Network Time Series app" window screen: "Evaluate Network"



Click in the "Train network" the "Next" Button. Feel free to re-evaluate the network to try to find better values with: "Train Again", "Adjust Network Size", "Import Larger Data Set", "Optionally perform additional tests", etc. Then, click the "Next Button".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 16 Description In the MATLAB "Neural Network Time Series app" window screen: "Deploy Solution" Select the option needed for your "Deploy Solution": In the section "Application Deployment" click button "MATLAB function," In "Code Generation" click button "MATLAB Matrix-Only Function," in "Simulink Deployment" click "Simulink Diagram" and in "Graphics" click button "Neural Network Diagram." Click "Next" button

16. In the MATLAB "Neural Network Time Series app" window screen: "Deploy Solution"

The screenshot shows the 'Deploy Solution' dialog box in MATLAB. It has four sections: 'Application Deployment', 'Code Generation', 'Simulink Deployment', and 'Graphics'. Each section has a 'Generate' button. Blue arrows with numbers 1 through 5 point to the following buttons: 1. 'MATLAB Function' in the Application Deployment section. 2. 'MATLAB Matrix-Only Function' in the Code Generation section. 3. 'Simulink Diagram' in the Simulink Deployment section. 4. 'Neural Network Diagram' in the Graphics section. 5. 'Next' button at the bottom right. At the bottom left, there are 'Neural Network Start' and 'Welcome' buttons. At the bottom center, there are 'Back' and 'Next' buttons. At the bottom right, there is a 'Cancel' button. A status bar at the bottom of the window reads 'From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa'.

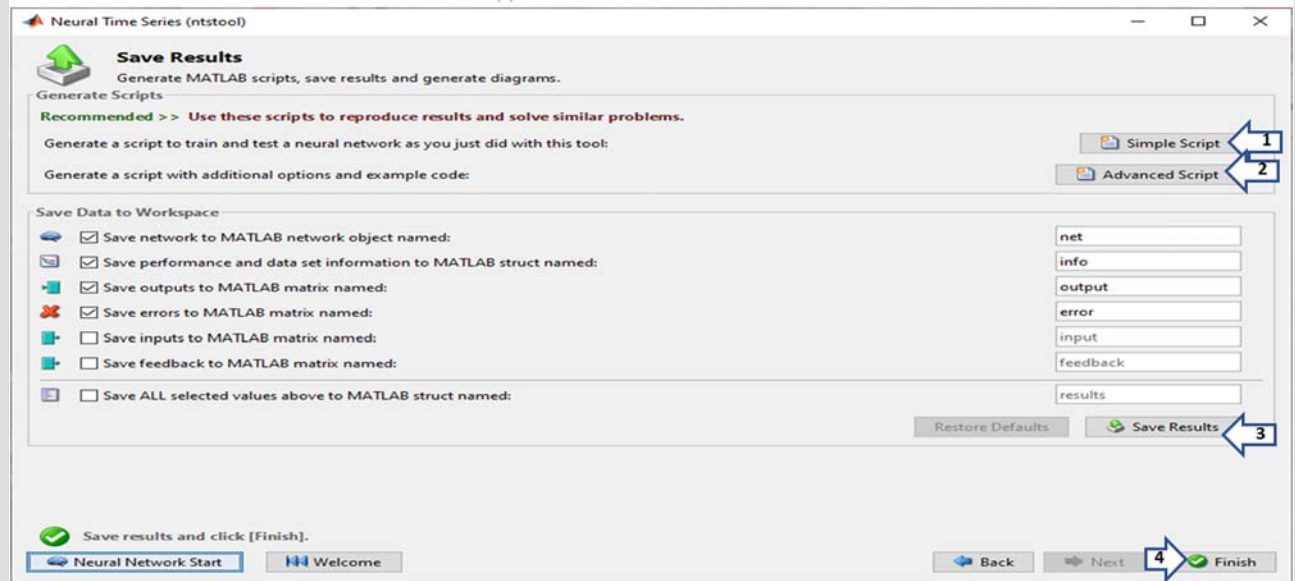
Select the option needed for your "Deploy Solution": In the section "Application Deployment" click button "MATLAB function", In "Code Generation" click button "MATLAB Matrix-Only Function", in "Simulink Deployment" click "Simulink Diagram" and in "Graphics" click button "Neural Network Diagram". Click "Next" Button.

17

In the MATLAB "Neural Network Time Series app" window screen: "Save Results"

In the section "Generate Scripts" click the buttons: "Simple Script" and "Advanced Script." In the section "Save Data to Workspace" click button "Save Results." Finally click on the button "Finish," and its confirmation dialog

17. In the MATLAB "Neural Network Time Series app" window screen: "Save Results"



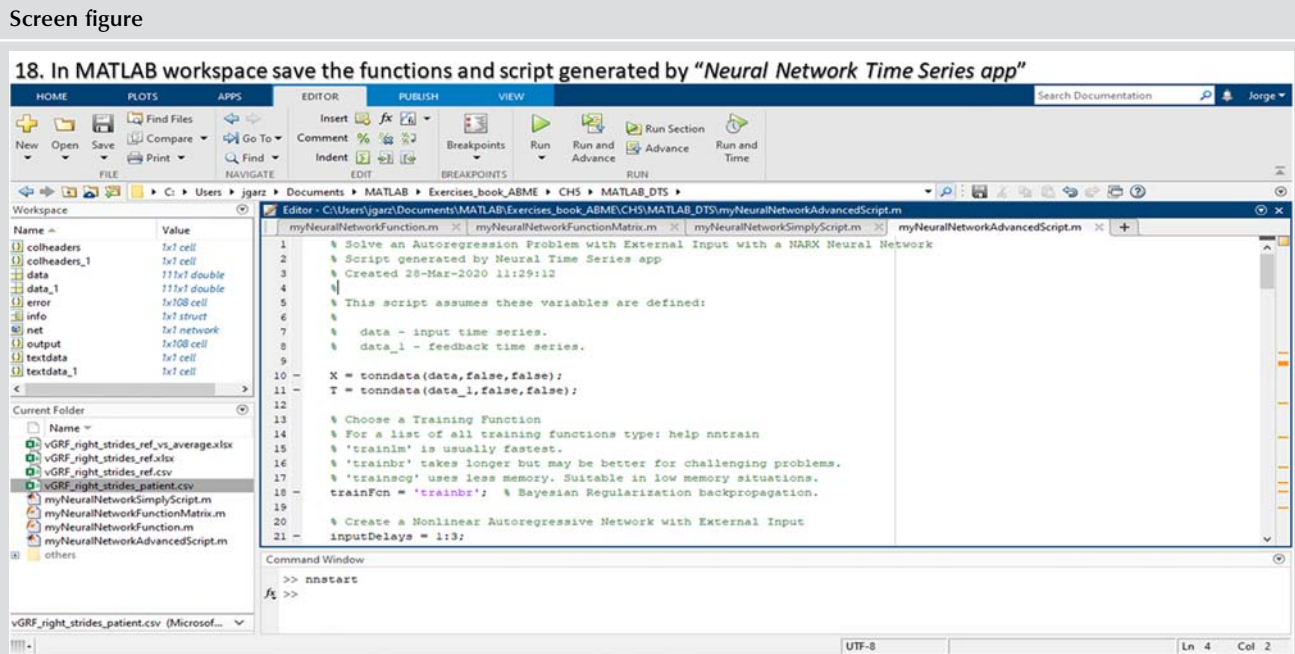
In the section "Generate Scripts" click the buttons: "Simple Script" and "Advanced Script". In the section "Save Data to Workspace" click button "Save Results". Finally click on the button "Finish", and its confirmation dialog.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 18 Description In MATLAB workspace save the functions and script generated by “Neural Network Time Series app” Save the two functions generated by MATLAB “Neural Fitting” as: “myNeuralNetworkFunction.m” and “myNeuralNetworkFunctionMatrix.m.” Save also the two scripts as: “myNeuralNetworkSimplyScript.m” and “myNeuralNetworkAdvancedScript.m.” Note: They can be useful for others projects



Save the two functions generated by MATLAB “Neural Fitting” as: “myNeuralNetworkFunction.m” and “myNeuralNetworkFunctionMatrix.m”. Save also the two scripts as: “myNeuralNetworkSimplyScript.m” and “myNeuralNetworkAdvancedScript.m”. Note : They can be useful for others projects...

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

19

Closing MATLAB tools and Plots generated by “Neural Network Time Series app”

Close without saving the “Simulink diagram for the ANN,” the “NARX Neural Network Map generated” and the “Neural Network—Neural Clustering App”

19. Closing MATLAB tools and Plots generated by “Neural Network Time Series app”

The screenshot shows the MATLAB Neural Network Start interface. The main window is titled "Neural Network Start (nnstart)" and displays a "Welcome to Neural Network Start" message. Below the message, there are sections for "Getting Started Wizards" and "More Information". The "Getting Started Wizards" section lists several wizards: "Fitting app (nftool)", "Pattern Recognition app (nprtool)", "Clustering app (nctool)", and "Time Series app (ntstool)". The "More Information" section provides details about each wizard and their capabilities.

The "NARX Neural Network (view)" window is also visible, showing a detailed diagram of the neural network architecture. It includes input nodes for $x(t)$ and $y(t)$, a hidden layer with 15 nodes, and an output layer with 1 node. The diagram shows the flow of information from the input nodes through the hidden layer to the output node, with weights W and biases b indicated.

Close without saving the “Simulink diagram for the ANN,” the “NARX Neural Network Map generated” and the “Neural Network – Neural Clustering App”.

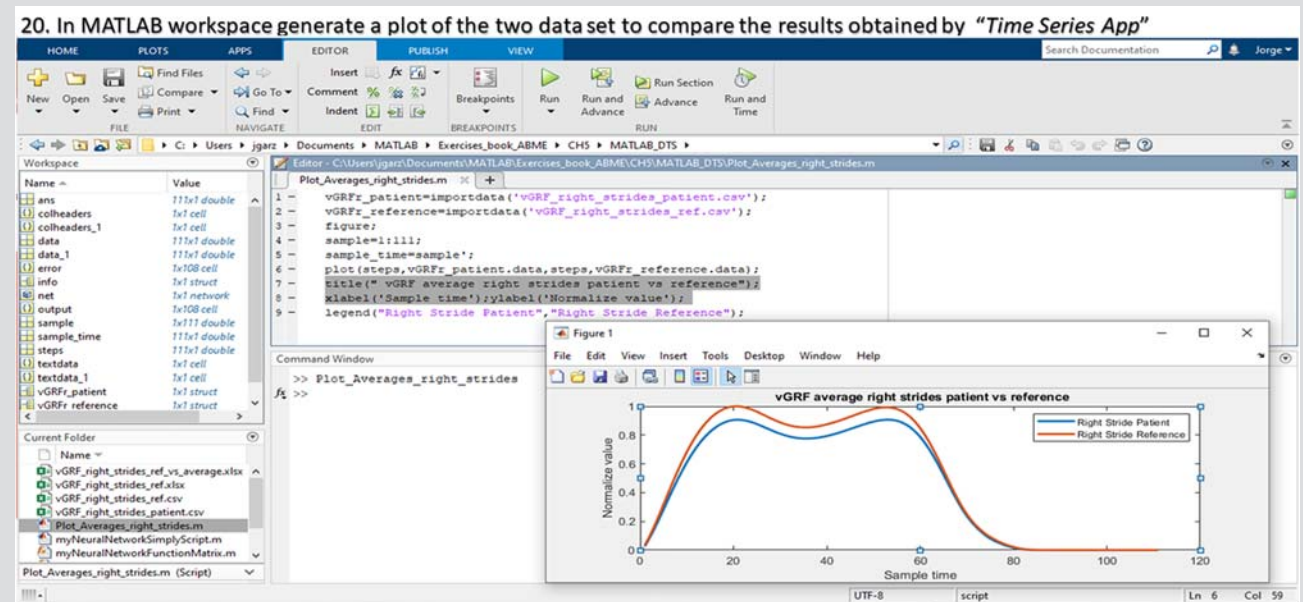
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 20 Description In MATLAB workspace generate a plot of the two dataset to compare the results obtained by "Time Series App"
Load the MATLAB script "Plot_Averages_right_strides.m," run the script that generate the two plot to observe the difference between them as were detected by the "NARX Neural Network Time Series app" as indicated in slide 14: the left plot of "Autocorrelation of error 1," and "Error Input-Error Cross-correlation" that indicates the differences between the "Target-Output." Finally, close all files and plots

Screen figure



Load the MATLAB script "Plot_Averages_right_strides.m," run the script that generate the two plot to observe the difference between them as were detected by the "NARX Neural Network Time Series app" as indicated in slide 14 : the left plot of "Autocorrelation of error 1," and "Error Input-Error Cross-correlation" that indicates the differences between the "Target - Output". Finally, close all files and plots. From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

An AI model from “MATLAB Deep Learning Toolbox” was obtained by defining, building, training, and deploying a “NARX” to analyze and differentiate between “two normalized average vGRF signals” from the right limb during “brisk walking”: one of “healthy people” used as a reference versus “unhealthy patients.” Plots were obtained showing the “Time-Series Response” as a plot of “Output and Target versus Time,” “Error Histogram,” “Error Autocorrelation,” and the “Error Input-Error Cross-correlation” that indicates the differences between the “Target and Output,” showing lower values in the output obtained from the time series vector of the “vGRF_right_strides_patients.csv” with respect to the time series reference vector of the “vGRF_right_strides_ref.csv.” Finally, a results comparison between the AI model from “Recurrent Neural Network (RNN)-NARX” versus “a normal time plot” shows the AI model “NARX precisely detects the differences between the two vGRF” datasets.

Recommendation

This kind of AI NARX model can be applied to analyze comparison between biomedical signals from different bioinstruments and other applications involved with variables through time.

“Shallow Neural Networks” have many applications in biomedical engineering, such as automating detection and localization of myocardial infarction using shallow and end-to-end deep neural networks [31], in the domain of reservoir characterization [32], point cloud occlusion recovery with shallow feed forward neural networks [33], and many others.

5.4 Backpropagation neural networks types

“Backpropagation neural networks” imply that the signal propagates from the input data forward through its parameters toward the decision, and then propagates information about the error in reverse. In this way it can adjust the parameter until finding the smallest error. Some frequently used examples of *Backpropagation neural networks* are shown in Fig. 1.9: *Auto Encoder (AE)*, *Variational Auto Encoder (VAE)*, *Denoising Auto Encoder (DAE)*, *Sparse Auto Encoder (SAE)*, *Deep Convolution Network (DCN) or ConvNet (CNN)*, *Deconvolutional Network (DN)*, *Deep Convolutional Inverse Graphics Network (DCIGN)*, *Generative Adversarial Network (GAN)*, *Deep Residual Network (DRN) or Deep ResNet*, and others.

The idea of backpropagation is the improving of the AI model by recalibrating the weights going back from the output to the hidden layers of the “ANN.” For each training iteration, the algorithm will compute this error and will recalibrate its parameter, so that during the next iteration the error will be reduced, thus that the output will get closer and closer to the real target. The most frequently used backpropagation algorithms

are “Levenberg–Marquardt backpropagation,” “Bayesian regularization backpropagation,” and “Gradient descent”:

- “Levenberg–Marquardt backpropagation” uses the “Levenberg–Marquardt optimization” which is the most widely optimization algorithm applied to solve nonlinear least squares problems. It finds the minimum value especially in least squares curve, represented by the following equation: $f(x) = \frac{1}{2} \sum_{j=1}^m r_j^2(x)$. “Levenberg–Marquardt optimization” is a blend of “gradient descent” and “Gauss–Newton iteration” solving the equation: $\min_f (x^k + \alpha_k p^k) \forall \alpha_k > 0$ [34]. In many cases it is fast and finds a solution even if it starts extremely far from the final minimum fitting, finding a local minimum which is not necessarily the global minimum.
- “Bayesian regularization backpropagation” updates the weight and bias values according to “Levenberg–Marquardt optimization.” It minimizes a combination of squared errors and weights, and then determines the correct combination to produce a network that generalizes well. Backpropagation is used to calculate the “Jacobian JX ” where the “Jacobian matrix” is the matrix of all first-order partial derivatives of net performance with respect to the weight and bias variable “ X .” Using the following equation: $JJ = JX * JX^T J_e = JX * EdX = -(JJ + I * mu) J_e$ [35]. This algorithm typically requires more time, but can result in good generalization for difficult, small, or noisy datasets. Training stops according to adaptive weight minimization (regularization).
- “Gradient Descent*” [36] has the goal of reduced the error toward zero, evaluating the “loss function” versus “neuron (nodes) weight” until finding the minimum value (ideally zero) before increasing again. Then the new weights are calculated using the following equation: $w'_i = w_i - \frac{\delta L}{\delta w_i}$. This equation will change the next weight (w'_i) to the first derivative of the loss (“ L ”) with respect to the actual weight (w_i).

“Gradient descent” and the “conjugate gradient method” are both algorithms for minimizing nonlinear functions, that is, functions like the “Rosenbrock function” as a nonconvex function or a “multivariate quadratic function” when there are more than one independent variables with quadratic relationship with the dependent variable.

5.4.1 Auto Encoder

An “Autoencoder Neural Network (AE)” is an unsupervised learning (feature learning) that has “Encoders” and

“Decoders”; they are “*feed forward NN*” but apply “*back-propagation algorithms*.” They are studied in this section for setting the target values to be equal to the inputs, that is, it uses “ $y(i) = x(i)$.” “*Auto Encoder (AE)*” is basically an unsupervised artificial neural network that learns how to efficiently compress and encode data from images, and how to reconstruct the data back from the reduced encoded representation to a representation that is as close to the original input as possible.

Then as a general view, an “*Autoencoder NN*” is composed of an “*encoder*” and a “*decoder*.” The “*encoder*” and “*decoder*” can have multiple layers, but for simplicity consider that each of them has only one layer. “*Autoencoders*” can be used as tools to learn “*deep neural networks*.” Training an “*autoencoder*” is unsupervised in the sense that no labeled data is needed. The training process is still based on the optimization of a cost function. The cost function measures the error between the input “ x ” and its reconstruction at the output “ $y = \hat{x}$.” “*Autoencoder*,” by design, reduces data dimensions by learning how to ignore the noise in data and images [37]. AE consists of four main parts, as shown in Fig. 5.11A: “*Encoder*,” “*Bottleneck*,” “*Decoder*,” and “*Reconstruction loss*”:

- “*Encoder*” is where the model learns to reduce input dimensions and compress input data using activation function as “*identity*” or “*sigmoid*,” as indicated in Fig. 5.11B.
- “*Bottleneck*” is a layer that contains the compressed representation of the input data of images.
- “*Decoder*” is where the model learns to reconstruct data from the encoded representation.

- “*Reconstruction loss*” is a method that measures the “*decoder values*” and outputs them to the original data using the equation “ $E = \text{Loss function during the training}$ ” shown in Fig. 5.11C.

Example 5.6: Example of Auto Encoder Neural Network to reconstruct chest X-rays.

Problem to resolve

There is a need to have an AI model that can compress and reconstruct “*normal chest X-rays*” and “*nonnormal chest X-rays affected by Coronavirus (COVID-19) pneumonia*” to begin “*research to analyze the chest X-ray images by AI models*.”

Proposed solution for reconstruct X-rays images using “Auto Encoders (AE)”

An “*Auto Encoder (AE) to reconstruct X-Ray images*” is proposed with a MATLAB script [38]. Open the MATLAB program and go to the book data companion directory:

“`...\Exercises_book_ABMENCH5\MATLAB_AE`” and open the script “`AE.m`.” This script is divided into four steps as shown in Fig. 5.12A). These steps are:

- *Step 1) “Read, prepare, and train X-ray images.”* The subdirectory “*Train*” has “*two X-ray images from normal chests*.” They are read and images are loaded as 1×2 cells, where each one is a “*m by n by 3 matrix*”. Each is then converted to “*black and white*” to a “*m by n logical matrix*”, and finally to a “*m by n double matrix*”, the original images and the same images converted to “*black and white*” are shown in the middle of Fig. 5.12 in the step 1) section. The resulting

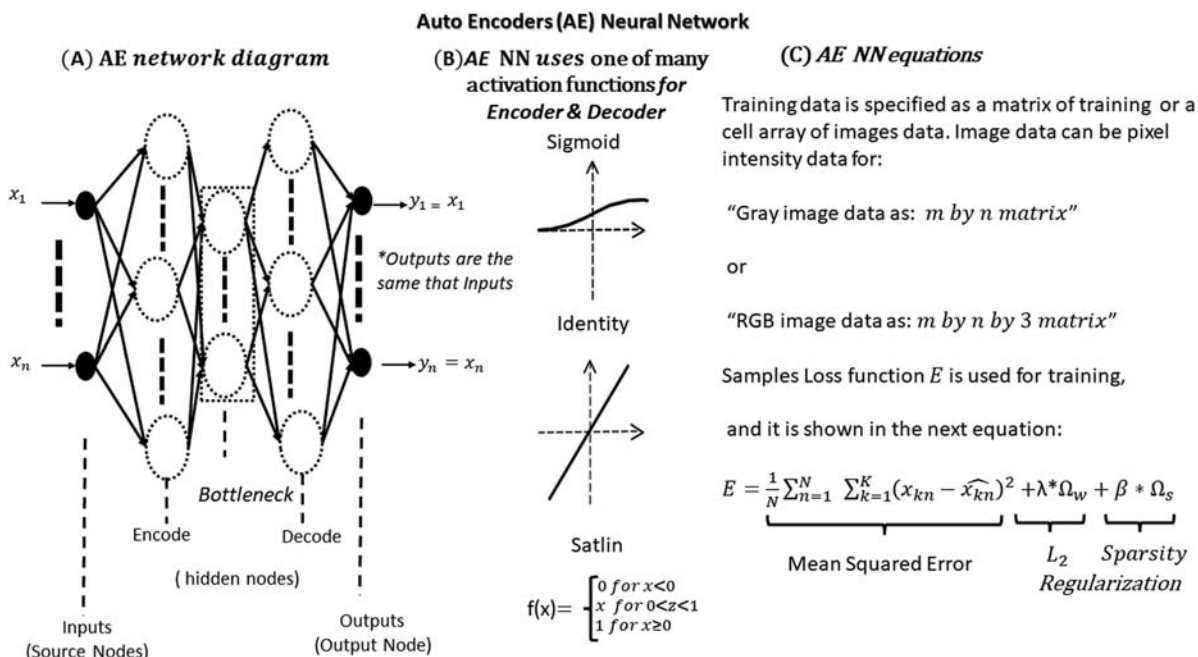


FIGURE 5.11 Auto Encoder (AE) Neural Network: (A) AE NN diagram, (B) AE NN activation functions, and (C) AE NN equations.

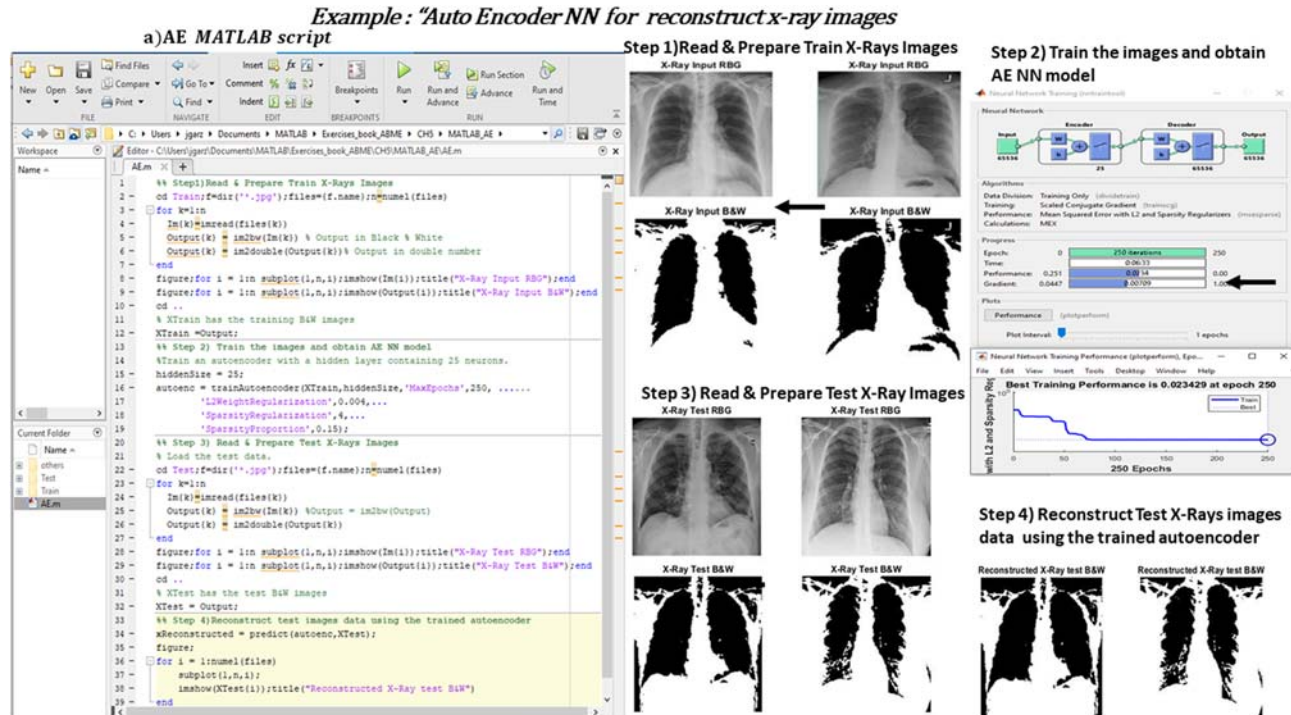


FIGURE 5.12 Example using "Auto Encoder NN for reconstruct X-ray images": a) AE MATLAB script, and steps 1) to 4) results.

cells are assigned to the "XTrain" variable to be used for the training.

- **Step 2) "Train the images and obtain AE NN model."** An "AE NN" is defined with 25 hidden neurons for "Encoder," that generate 65536 neurons for the "Decoder" based in the image size of 256×256 pixels. Note: The "m by n by 3 matrix" can also be trained but it will require $256 \times 256 \times 3 = 196,608$ neurons in the "Decoder," besides the training takes approx. three times or more longer in terms of processing time. The "Performance" is shown to be verified if 250 epochs are enough to lower the error, as shown at top right side of Fig. 5.12.
- **Step 3) "Read and prepare test X-ray images."** The subdirectory "Test" has two X-ray images: one "normal chest X-rays" and another "nonnormal chest X-ray affected by Coronavirus (COVID-19) pneumonia." The images are read and are loaded as a 1×2 cell, where each one is a "m by n by 3 matrix", which is then is converted to "black and white," as shown in the bottom middle section in Fig. 5.12. The images are converted into a "m by n logical matrix", and finally to a "m by n double matrix". The resulting cells are assigned to the "XTest" variable.
- **Step 4) "Reconstruct test images data using the trained autoencoder."** The "XTest" variable is used in the reconstruction through the prediction model stored in the variable "autoenc." And the final reconstructions are shown in the right bottom of Fig. 5.12.

Conclusions

The basic AE model obtained in this example shows that is possible to compress and reconstruct "normal chest X-rays" and "nonnormal chest X-rays affected by Coronavirus (COVID-19) pneumonia X-rays" in order to begin "research to analyze the chest X-ray images by AI models."

"AE" has helped to resolve many problems in biomedical engineering such as multikernel fuzzy clustering based on an auto-encoder for fMRI functional network (fMRI brain networks images) [39], predicting drug–drug interactions using multimodal deep auto-encoders-based network embedding and positive-unlabeled learning [40], cross-modal guidance-based auto-encoder for multivideo summarization [41], and many others.

5.4.2 Variational Auto Encoder

"Variational Auto Encoders (VAE)" are powerful generative models that have many different applications, such as generating fake human faces, fake biomedical images to be used in research and comparative methods, and many others. In the standard "AE," the input is connected to a "encoder," then data is "compressed" and finally it is connected to a "decoder." Its objective is to replicate the same image from the input in the output. In "generative AI models," the objective is generating variations in the output

from an input image. We can define an “Encoder” and “Decoder” as a two-neural network, where the “encoder” has an input of x_n data points and its output has a hidden representation of $z_{h,v}$, where “ $h = \text{number of horizontal pixels}$ ” and “ $v = \text{number of vertical pixels}$ ” composed of its “weights $w_{i,j}$ ” and its “bias \emptyset ”. In the example of Fig. 5.12, the last section the X-ray images had 256×256 pixels of resolution, so each image is encoded to 65,536 in the representation of $z_{h,v}$. Then, the “encoder” must learn to handle the data of this “bottleneck” represented as $E_{\emptyset}(z_{h,v}|x_n)$ that represents a “Gaussian Probability Density” as shown in the inferior part of the Fig. 5.13A. In the “VAE” samples from this distribution can be altered for the representation of “ $z_{h,v}$ ” to “ $z_{h,v}^g$ ”. And the “decoder” as another neural network has input data points and its outputs are the parameters of the probability function with their “weights” and “bias” that can be presented as $p_{\emptyset}(y_n|z_{h,v}^g)$, where the probability distribution for each hidden neuron that represent a pixel can be represented using a “Bernoulli distribution” that can be “0” or “1.”

The Loss function of the “VAE” is the “mean squared error” as a negative log-likelihood plus a “regularizer” indicated as “ L_2 ”, plus Sparsity”S”.

“VAEs” help to resolve problems in biomedical engineering such as variational graph auto-encoders for (Continued)

(Continued)

miRNA-disease association prediction [42], matrix-variate variational auto-encoder with applications to image process [43], multitask learning using variational auto-encoder for sentiment classification [44], and many others.

5.4.3 Denoising Auto Encoder

“Denoising Auto Encoder (DAE)” solves the problem known as “Identify Function*” by corrupting the data on purpose by randomly turning some of the input values to zero. In general, the percentage of input nodes which are being set to zero is about 50%. When calculating the “Loss function” the more important thing is to compare the output values with the original input values, not with the corrupted input. This way, the risk of learning the identity function instead of extracting features is eliminated [45].

In the example shown in Fig. 5.13B, the “DAE” inputs: x_2, x_4 , and x_7 are altered (corrupted) to “0,” then the risk of learning with the “identity function” is avoided.

Note*: Identify Function or “Null Function” is present in neural network when there are more nodes in the hidden layer than there are in the inputs, and it could skip learning this important function.

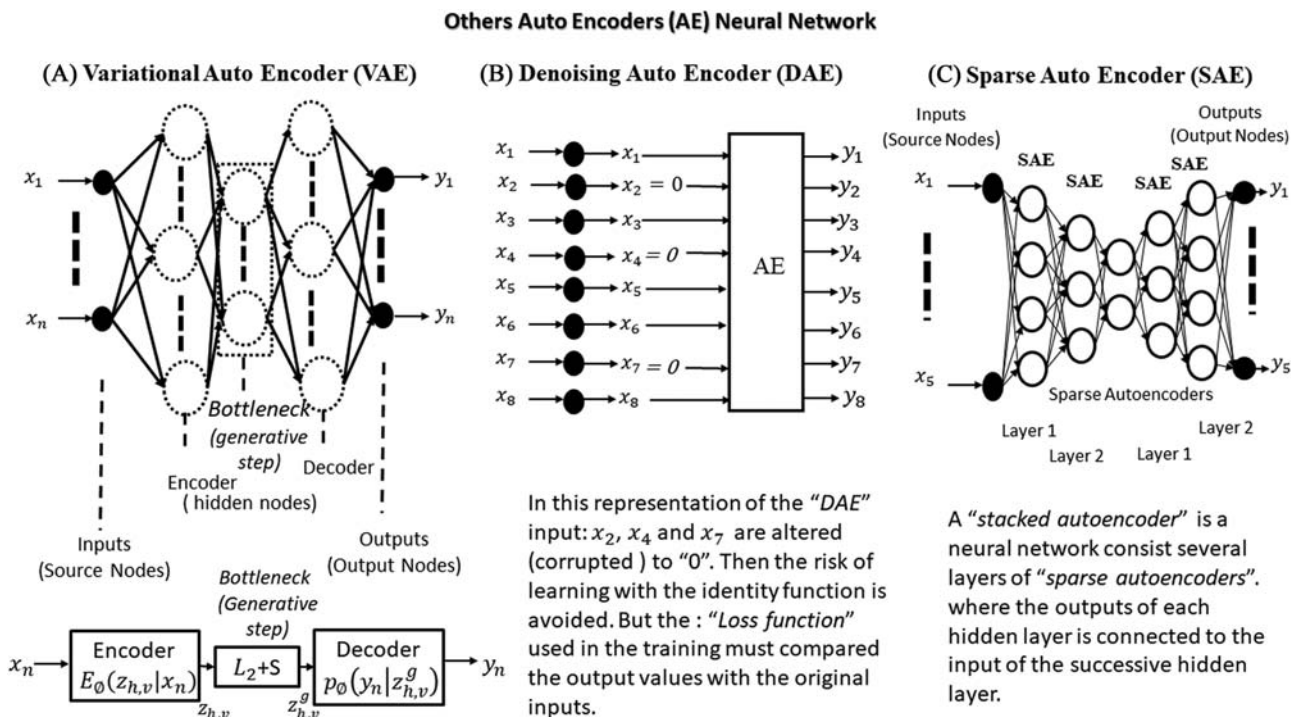


FIGURE 5.13 Other Auto Encoders (AE) Neural Network: (A) Variational Auto Encoder (VAE), (B) Denoising Auto Encoder (DAE), and (C) Sparse Auto Encoder (SAE).

“DAEs” can be used in numerous problems of biomedical engineering such as novel performance prediction model of a biofilm system treating domestic wastewater based on stacked denoising auto-encoders deep learning network [46], ECG signal enhancement based on improved denoising auto-encoder [47], convolutional auto-encoder for image denoising of ultralow-dose CT [48], and many others.

5.4.4 Sparse Auto Encoder and stacked auto encoders

“Sparse Auto Encoder (SAE)” consists of a single hidden layer, which is connected to the input vector by a weight matrix forming the encoding step. The hidden layer then outputs to a reconstruction vector, using a tied weight matrix to form the decoder [49]. “Stacked Auto Encoders” is a neural network consisting of several layers of sparse autoencoders where the output of each hidden layer is connected to the input of the successive hidden layer [50]. A “stacked autoencoder” is a neural network consisting of several layers of “sparse autoencoders” where the output of each hidden layer is connected to the input of the successive hidden layer, as shown in Fig. 5.13C, where are four “sparse autoencoders” to improve accuracy in deep learning with noisy autoencoders embedded in the layers.

“Stacked autoencoders” are used for the diagnosis of breast cancer with stacked autoencoder and Subspace kNN [51], text feature extraction based on stacked variational autoencoder [52], classification and diagnosis of cervical cancer with stacked autoencoder and softmax classification [53], and many more Biomedical Engineering solutions.

5.4.4.1 Research 5.4* Backpropagation Neural Network for Patterns Recognition and classification of “Breast Cancer”

Note*: This research uses a “Shallow neural network” to explain concepts that are necessary to understand for the next study in Research 5.5, Deep Learning using a “Pretrained Deep Convolutional Neural Network to obtain an AI model to classify Mammograms standard views types.”

5.4.4.1.1 Case for research

“Apply MATLAB Deep Learning Toolbox” using a Backpropagation Neural Network for Pattern Recognition and classification to obtain an AI model for breast cancer.”

5.4.4.1.2 General objective

Apply “MATLAB Deep Learning Toolbox” to define, build, train, and deploy a “Backpropagation Neural Network” for

“Patterns Recognition and classification” to obtain an “AI model for breast tumors,” used for medical diagnostic based on nine input variables of physical tumor dimensions.

5.4.4.1.3 Specific objectives

- Load the “Cancer.csv” dataset for input of the “backpropagation ANN,” and its “Cancer_target.csv” file as the data to be used to obtain the AI model.
- Define validation and testing dividing the samples randomly as: “70% for training,” “15% for validation,” and “15% for testing.”
- Select the optimal number of “Hidden layers” for the best “Patterns Recognition and classification.”
- Train the “backpropagation ANN” applying the “Conjugate Gradient BackPropagation” algorithm for the “Patterns Recognition and classification.”
- Learn the best parameters criteria to detect the best performance model based on: “Confusion matrix” and “ROC” plot.
- Deploy the MATLAB “Patterns Recognition and classification” AI model.
- Test the MATLAB “Patterns Recognition and classification” model as a function matrix based on nine input tumor measurement variables for classifying the “breast tumor” as: “Benign cancer” or “Malignant cancer” using probability scores.

5.4.4.1.4 Background for “Breast cancer”

Breast tumors are defined as a “mass of abnormal tissue.” There are two types of “breast cancer tumors”: “benign cancer” and “malignant cancer” [54]:

- “Benign cancer” is nonaggressive tumors that do not affect surrounding tissues; occasionally they may continue to grow, pressing on organs and causing pain or other kind of problems; they usually are removed.
- “Malignant cancer” is aggressive cancerous tumors, that invade and damage surrounding tissues. They are usually analyzed by performing a biopsy to determine the severity of the tumor. The malignant cancer tumor spreads to others parts of the body, usually through the lymph system, and forms secondary tumors.

There are two frequently used ways to classify “breast cancer tumors”: “breast tumor grading” and “breast tumor stage” [55]:

- “Breast tumor grading” is made when the tumor is removed from the breast, and three cancer cell features are studied and each is assigned a score in the pathology report as:
 - “Grade 1 or well differentiated,” when the cancer cells are slower growing, look more like normal breast tissue and the “score assigned could be from 3 to 5.”
 - “Grade 2 or moderately differentiated,” when the cancer cells do not look like normal cells, growing

and dividing a little faster than normal and the “score assigned could be 6 or 7.”

- “Grade 3 or poorly differentiated,” when the cancer cells look very different from normal cells, will probably grow faster, and spread faster, and the “score assigned could be 8 or 9.”

Note: Having a low-grade cancer is an encouraging sign. But higher-grade cancers may be more vulnerable than low-grade cancers to treatments such as chemotherapy and radiation therapy that work by targeting fast-dividing cells.

- “Breast tumor stage” describes how much the cancer has spread in the human body [56]. The earliest stage breast cancers are “stage 0 (carcinoma in situ).” It then ranges from “stage I (1) through IV (4).” As a rule, “the lower the number, the less the cancer has spread. A higher number, such as stage IV, means cancer has spread more. And within a stage, an earlier letter means a lower stage.”

5.4.4.1.5 Dataset

The dataset used for “breast cancer tumor” that is useful for pathologists follows the features of biopsies via “Fine

Needle Aspiration (FNA)” in accordance with Wisconsin grade scale based on nine cytological characteristics of breast FNAs. They are valued on a scale of 1 to 10, with 1 being the closest to benign and 10 the most anaplastic [57] to determine whether a “breast mass is benign or malignant.” The dataset has been divided into two files: “Cancer.csv” with information from 699 instances with nine fields (attributes), as indicated in Table 5.9, and another file named “Cancer_targets,” as shown in Table 5.10.

Note: This data is available from the UCI Machine Learning Repository (<http://mlearn.ics.uci.edu/MLRepository.html>) [58].

5.4.4.1.6 Procedure

The steps to obtain an AI model “using MATLAB Deep Learning Toolbox” using a backpropagation neural network for patterns recognition and classification for breast cancer” are summarized in Table of slides 5.4 and each step of the example is visually explained using screen sequences with instructions in easy to follow screen figures.

TABLE 5.9 Dataset “Cancer.csv” fields and descriptions.

Field	Description *699 instances of cancer tumors. Number of fields = 9
Clump_thickness	Clump thickness (normalized decimal from 0.1 to 1)
Uniformity_cell_size	Uniformity of cell size (normalized decimal from 0.1 to 1)
Uniformity_cell_shape	Uniformity of cell shape (normalized decimal 0.1 to 1)
Marginal_adhesion	Marginal adhesion (normalized decimal from 0.1 to 1)
Single_epithelial_cell_size	Single epithelial cell size (normalized decimal from 0.1 to 1)
Bare_nuclei	Bare nuclei (normalized decimal from 0.1 to 1)
Bland_chomatin	Bland chomatin (normalized decimal from 0.1 to 1)
Normal_nucleoli	Normal nucleoli (Normalized decimal from 0.1 to 1)
Mitoses	Mitoses (normalized decimal from 0.1 to 1)

Note: This dataset is also included in the companion directory of the book, in the following directory: “...\Exercises_book_ABME\CH5\MATLAB_PR\Cancer.csv.”

TABLE 5.10 Dataset for “Cancer_targets” for the Dataset “Cancer.csv.”

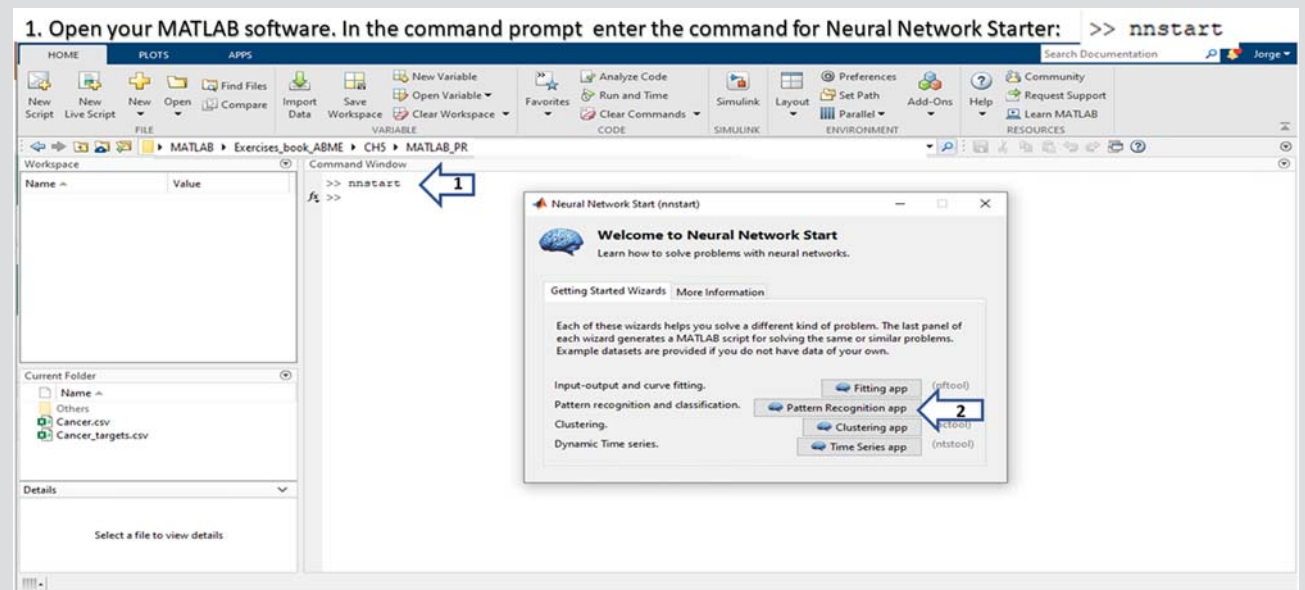
Field	Description *699 instances of cancer tumors. Number of fields = 2
Benign_cancer	Benign cancer ([0 = no, 1 = yes])
Malignant_cancer	Malign cancer ([0 = no, 1 = yes])

Note: This dataset is included in the companion directory of the book, in the following directory: “...\Exercises_book_ABME\CH5\MATLAB_PR\Cancer_targets.csv.”

Table of slides 5.4 Steps to obtain an AI model “MATLAB Deep Learning Toolbox using a Neural Network Patterns Recognition and classification for “breast cancer.”

Slide 1 Description
 Open your MATLAB software. In the command prompt enter the command for Neural Network Starter >> nnstart “MATLAB Deep Learning Toolbox” includes a “Neural Network GUI – nnstart” that opens a window with launch buttons for “Neural Network Fitting,” “Pattern Recognition,” “Clustering” and “Time series tools.” Click the button: “Pattern Recognition App (nprtool)”

Screen figure



“MATLAB Deep Learning Toolbox™” includes a “Neural Network GUI – nnstart” that opens a window with launch buttons for “Neural Network Fitting”, “Pattern Recognition”, “Clustering” and “Time series tools”. Click the button: “Pattern Recognition App (nprtool)”

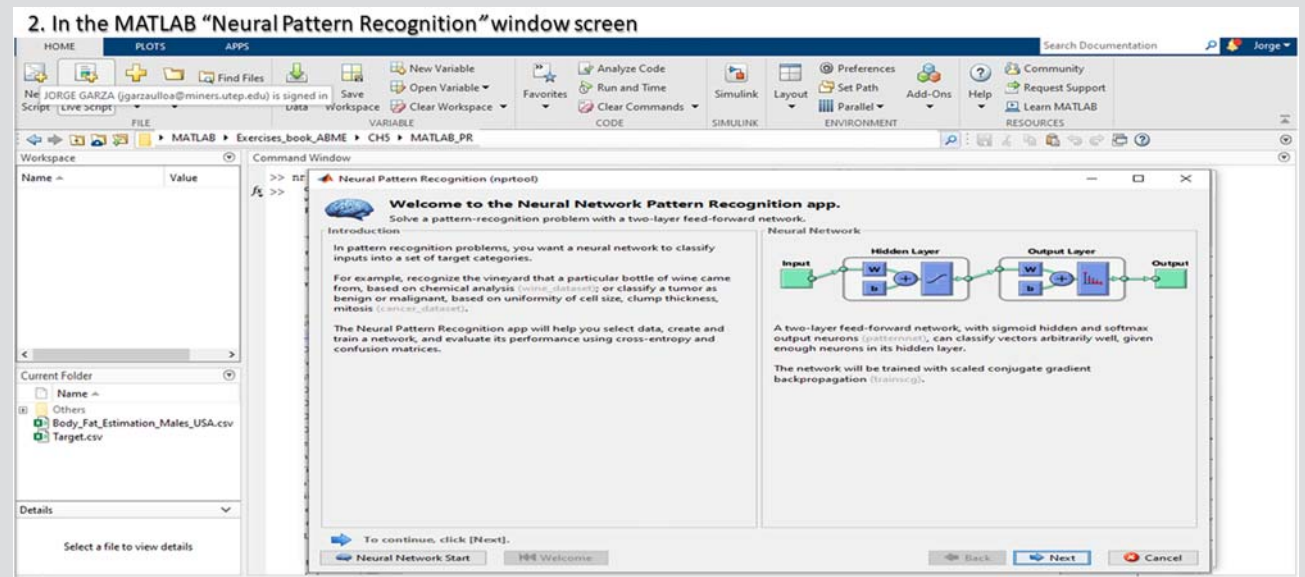
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 2 Description In the MATLAB "Neural Pattern Recognition" window screen "Neural Network Pattern Recognition" is used when there is a need to classify a "dataset of numeric inputs" with a numerical target using "Neural and training using a conjugate gradient backpropagation" algorithm, with a specified "Sigmoid hidden neurons" and a "Softmax output neurons." Click the "Next" button

Screen figure



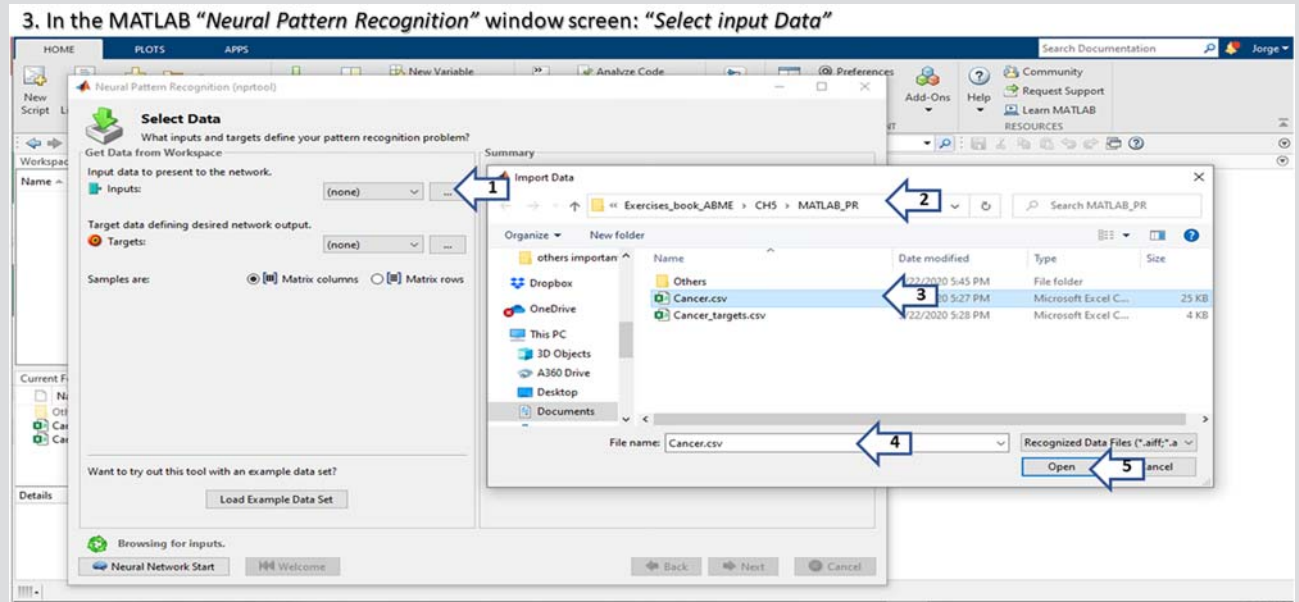
"Neural Network Pattern Recognition" is used when there is a need to classify a "dataset of numeric inputs" with a numerical target using "Neural and training using a conjugate gradient backpropagation" algorithm , with an specified "Sigmoid hidden neurons" and a "Softmax output neurons" .Click the "Next" button

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

3

In the MATLAB "Neural Pattern Recognition" window screen: "Select input Data"

Select the button for "input dataset ...," then select the file located at "... \Exercises_book_ABME\CH5 \MATLAB_PR\Cancer.csv," and click the button "Open"



Select the button for "input dataset ...", then select the file located at "... \Exercises_book_ABME\CH5\MATLAB_PR\Cancer.csv", and click the button "Open"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

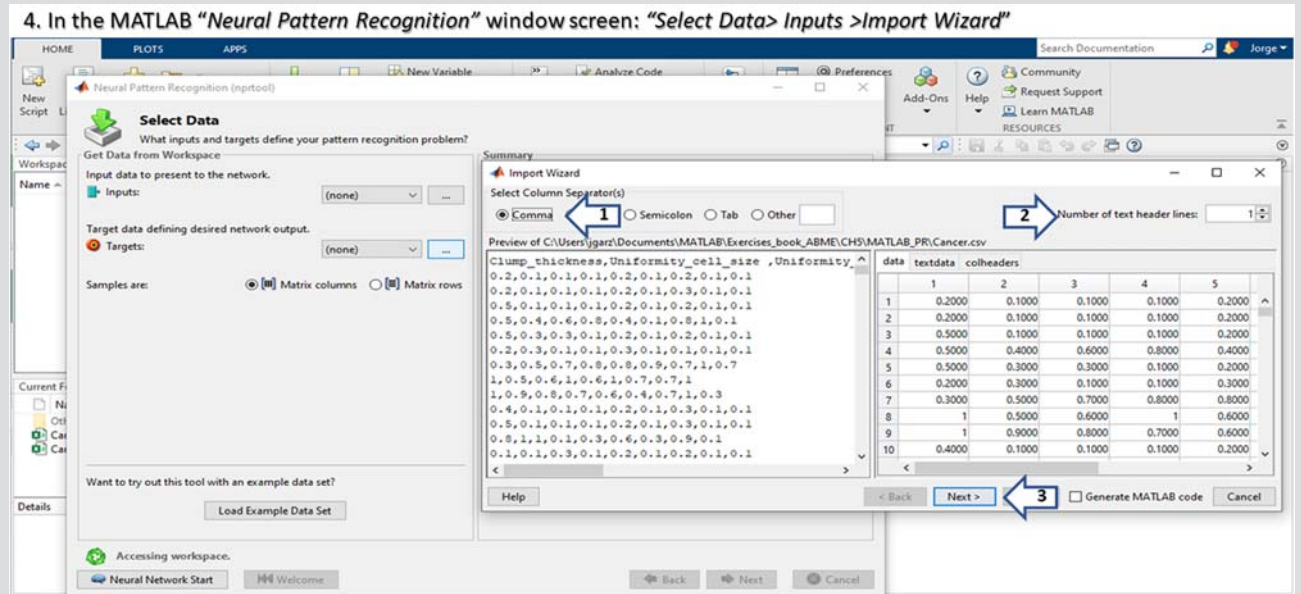
(Continued)

(Continued)

Slide Description

Screen figure

4 In the MATLAB "Neural Pattern Recognition" window screen: "Select Data > Inputs > Import Wizard" Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" button



Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" Button

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

5

In the MATLAB "Neural Pattern Recognition" window screen: "Select Data > Inputs > Select Data"
In the section "Select variable to import using checkboxes" activate "Create variables matching preview," verify the creation of the three variables and click the "Finish" button

5. In the MATLAB "Neural Pattern Recognition" window screen: "Select Data > Inputs > Select Data"

Summary

Import Wizard

Select variables to import using checkboxes

- Create variables matching preview
- Create vectors from each column using column names.
- Create vectors from each row using row names.

Variables in C:\Users\jgarza\Documents\MATLAB\Exercises_book_ABME\CH5\MATLAB_PR\Cancer.csv

Import	Name	Size	Bytes	Class
<input checked="" type="checkbox"/>	colhead...	1x9	1318	cell
<input checked="" type="checkbox"/>	data	699x9	50328	double
<input checked="" type="checkbox"/>	textdata	1x9	1318	cell

No variable selected for preview.

Help < Back Next > **Finish** Create MATLAB code Cancel

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

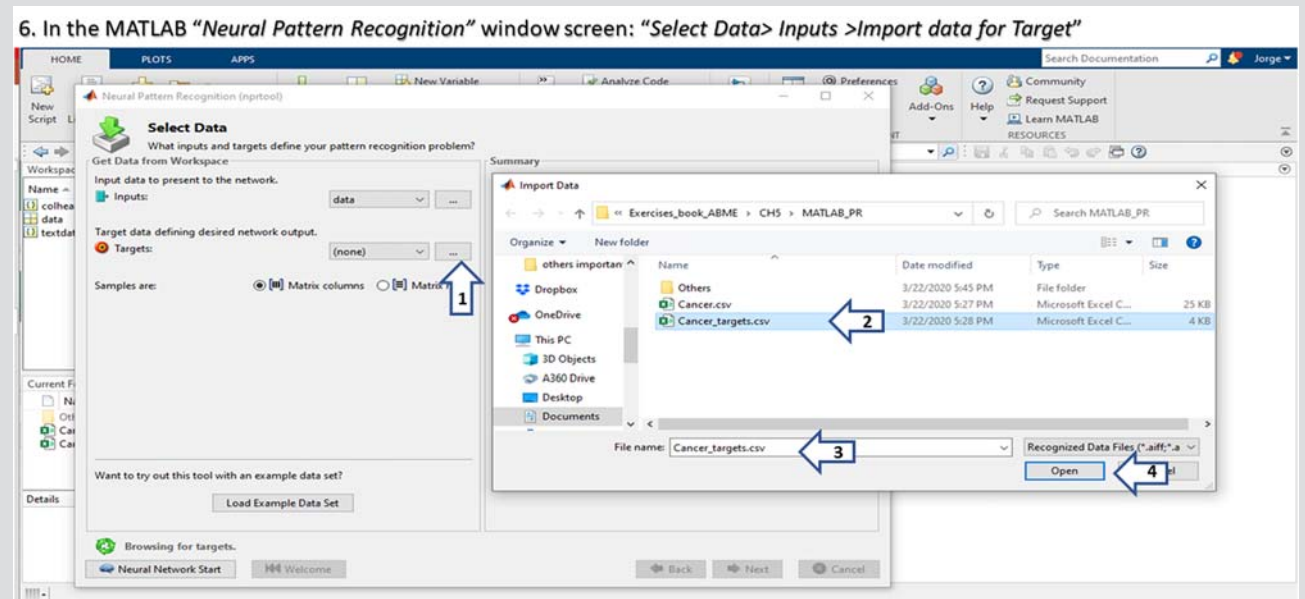
(Continued)

(Continued)

Slide Description

Screen figure

6 In the MATLAB "Neural Pattern Recognition" window screen: "Select Data > Inputs > Import data for Target"
Select the button for select the "target data ...," select the file "...\Exercises_book_ABME\CH5\MATLAB_PR\Cancer_targets.csv," and click the button "Open"

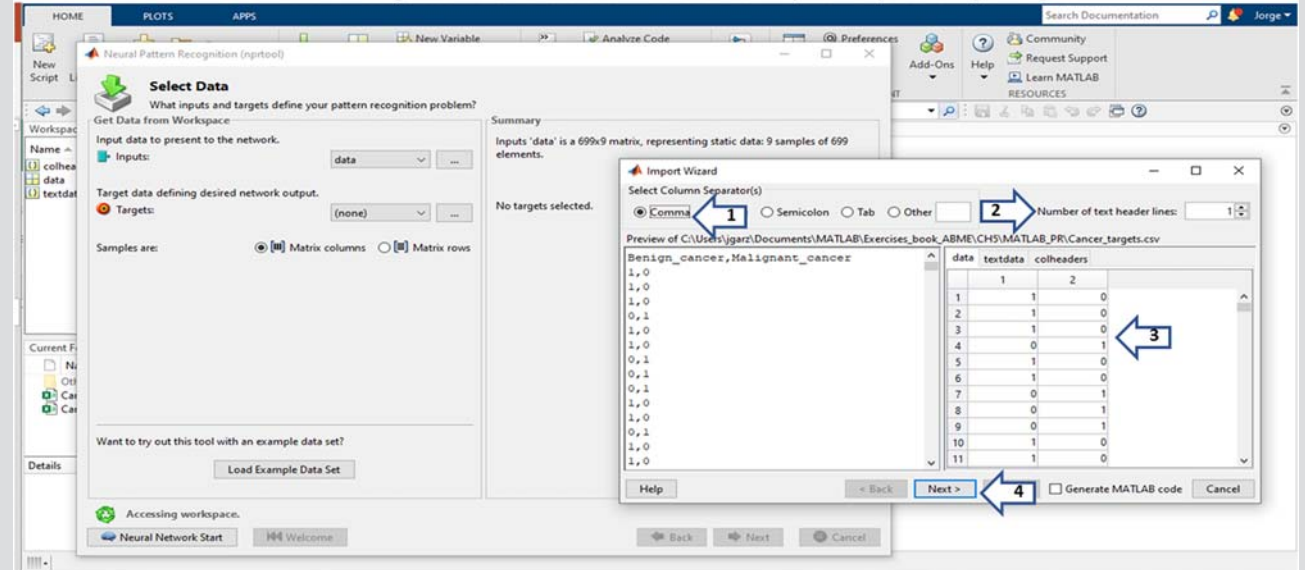


Select the button for select the "target data ...", select the file "...\Exercises_book_ABME\CH5\MATLAB_PR\Cancer_targets.csv", and click the button "Open"

7

In the MATLAB "Neural Pattern Recognition" window screen: "Select Data > Inputs > Import data for Targets"
Select "Comma" as the column separator, verify that the "Number of test header line is 1," and click on the "Next" button

7. In the MATLAB "Neural Pattern Recognition" window screen: "Select Data > Inputs > Import data for Targets"



Select "Comma" as the column separator, verify that the "Number of test header line is 1", and click on the "Next" Button

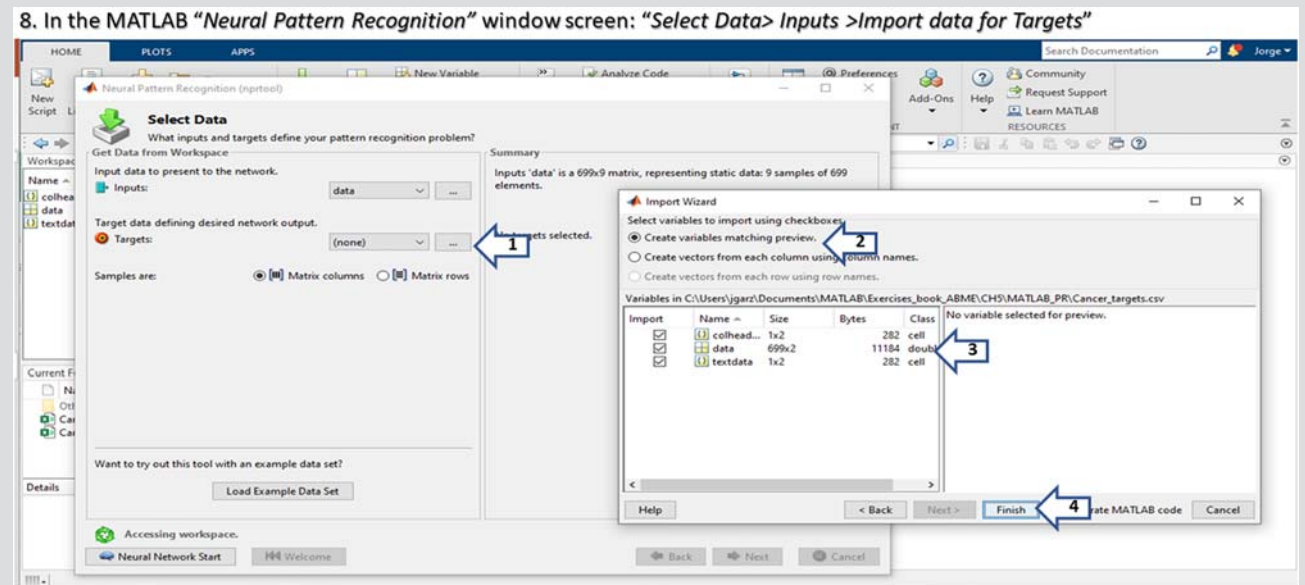
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Uflla

(Continued)

(Continued)

Slide 8 Description
In the MATLAB "Neural Pattern Recognition" window screen: "Select Data> Inputs >Import data for Targets"
In the section "Select variable to import using checkboxes" activate "Create variables matching preview," verify the creation of the three variables and click the "Finish" button

Screen figure



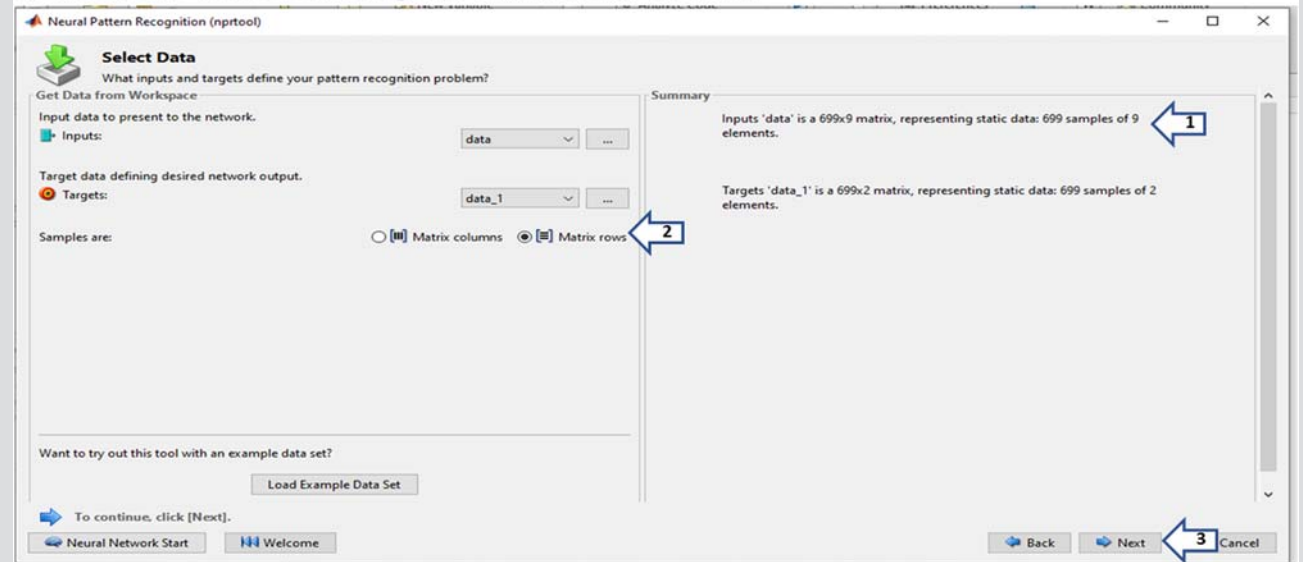
In the section "Select variable to import using checkboxes" activate "Create variables matching preview", verify the creation of the 3 variables and click the "Finish" button.

9

In the MATLAB “Neural Pattern Recognition” window screen: “Select Data”

In the “Summary” section verify that the “input data is 699×9 matrix,” specify the target as “Matrix rows,” check that the “Targets data is a 699×2 matrix,” and click in the button “Next”

9. In the MATLAB “Neural Pattern Recognition” window screen: “Select Data”



“In the “Summary” section verify that the “input data is 699×9 matrix”, specify the target as “Matrix rows”, check that the “Targets data is a 699×2 matrix”, and click in the button “Next”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

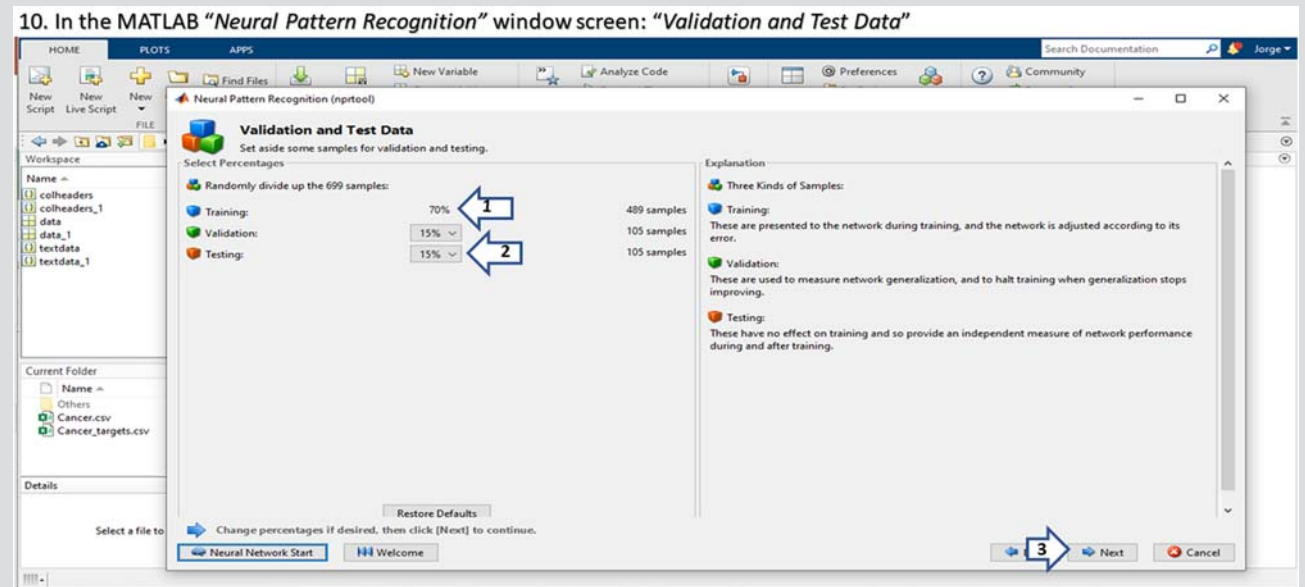
(Continued)

(Continued)

Slide Description

Screen figure

10 In the MATLAB "Neural Pattern Recognition" window screen: "Validation and Test Data" In the section "Select Percentages" verify that: "Training = 70%," "Validation = 15%," and "Testing = 15%," then click in the button "Next"



In the section "Select Percentages" verify that: "Training=70%," "Validation=15%" and "Testing=15%," then click in the button "Next".

11

In the MATLAB "Neural Pattern Recognition" window screen: "Network Architecture"

In the "Hidden Layer" section, define a "Neural Pattern Recognition" with "10 hidden neurons," in the "Neural Network" section verify that the "ANN" is defined with: "input of 9 variables," "Hidden Layers = 1 with 10," and an "Output layer = 1 with 2 variables," and 2 outputs. Then click in the button "Next"

11. In the MATLAB "Neural Pattern Recognition" window screen: "Network Architecture"

Neural Pattern Recognition (nprtool)

Network Architecture

Set the number of neurons in the pattern recognition network's hidden layer.

Hidden Layer

Define a pattern recognition neural network. (patternnet)

Number of Hidden Neurons:

Restore Defaults

Recommendation

Return to this panel and change the number of neurons if the network does not perform well after training.

Neural Network

Input: 9

Hidden Layer: 10

Output Layer: 2

Output: 2

Change settings if desired, then click [Next] to continue.

Neural Network Start Welcome Back Next Cancel

In the "Hidden Layer" section, define a "Neural Pattern Recognition" with "10 hidden neurons", in the "Neural Network" section verify that the "ANN" is defined with: "input of 9 variables", "Hidden Layers=1 with 10", and an "Output layer=1 with 2 variables", and 2 outputs. Then click in the button "Next".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

12 In the MATLAB "Neural Pattern Recognition" window screen: "Train Network"
In the "Train Network" the "scaled conjugate gradient backpropagation algorithm" is used, the training stop automatically stop when generalization stop improving based on an increase in the "Cross-entropy error" of the validation samples. Then click in the "Train" button

12. In the MATLAB "Neural Pattern Recognition" window screen: "Train Network"

	Samples	CE	%E
Training:	489	-	-
Validation:	105	-	-
Testing:	105	-	-

In the "Train Network" the "scaled conjugate gradient backpropagation algorithm" is used, the training stop automatically stop when generalization stop improving based on an increase in the "Cross-entropy error" of the validation samples. Then click in the "Train" button.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

13

In the MATLAB “Neural Pattern Recognition” window screen: “Neural Network Training—Conjugate Gradient Backpropagation” Observe “Cross-entropy (CE)” for Training, Validation, and Testing, keep clicking “Retrain” until you detect lower values for “CE” (Note: The values can be different, depend of the initial random value). Click the “Plot Confusion” to see the correct and incorrect percentages,” for training a 96.7% were correct and 3.3% incorrect

13. In the MATLAB “Neural Pattern Recognition” window screen: “Neural Network Training - Conjugate Gradient Backpropagation”

The screenshot displays the MATLAB Neural Network Training window. The main window is titled "Neural Network Training - Conjugate Gradient Backpropagation". It shows the training progress and results. The "Train Network" section indicates that the network is trained using scaled conjugate gradient backpropagation (trainscg). A "Retrain" button is visible. The "Results" section shows the following values:

	Samples	CE	%E
Training:	489	7.22357e-1	3.27198e-0
Validation:	105	1.95507e-0	2.85714e-0
Testing:	105	1.93794e-0	9.52380e-1

The "Confusion (plotconfusion)" window is open, showing four confusion matrices:

- Training Confusion Matrix:**

Output Class 1	305	7	97.8%
Output Class 2	9	168	94.9%
Target Class 1	62.4%	1.4%	2.2%
Target Class 2	1.8%	34.4%	5.1%
Target Class 3	97.1%	96.0%	96.7%
Target Class 4	2.9%	4.0%	3.3%
- Validation Confusion Matrix:**

Output Class 1	66	0	100%
Output Class 2	3	36	92.3%
Output Class 3	2.9%	34.3%	7.7%
Output Class 4	95.7%	100%	97.1%
Output Class 5	4.3%	0.0%	2.9%
Target Class 1	62.9%	0.0%	0.0%
Target Class 2	2.9%	34.3%	7.7%
Target Class 3	95.7%	100%	97.1%
Target Class 4	4.3%	0.0%	2.9%
- Test Confusion Matrix:**

Output Class 1	74	0	100%
Output Class 2	1	30	96.8%
Output Class 3	1.0%	28.6%	3.2%
Output Class 4	98.7%	100%	99.0%
Output Class 5	1.3%	0.0%	1.6%
Target Class 1	70.5%	0.0%	0.0%
Target Class 2	1.0%	28.6%	3.2%
Target Class 3	98.7%	100%	99.0%
Target Class 4	1.3%	0.0%	1.6%
- All Confusion Matrix:**

Output Class 1	445	7	98.5%
Output Class 2	13	234	94.7%
Output Class 3	1.9%	33.5%	5.3%
Output Class 4	97.2%	97.1%	97.1%
Output Class 5	2.8%	2.9%	2.9%
Target Class 1	63.7%	1.0%	1.5%
Target Class 2	1.9%	33.5%	5.3%
Target Class 3	97.2%	97.1%	97.1%
Target Class 4	2.8%	2.9%	2.9%

The "Notes" section contains the following text: "Training multiple times will generate different results in good classification. means no error. action of samples which are bans no misclassifications, classifications."

At the bottom of the window, there are buttons for "Neural Network Start", "Welcon", "Back", "Next", and "Cancel".

Observe “Cross-entropy (CE)” for Training, Validation and Testing, keep clicking “Retrain” until you detect lower values for “CE” (Note: The values can be different, depend of the initial random value). Click the “Plot Confusion” to see the correct and incorrect percentages” e.g. for training a 96.7% were correct and 3.3% incorrect.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

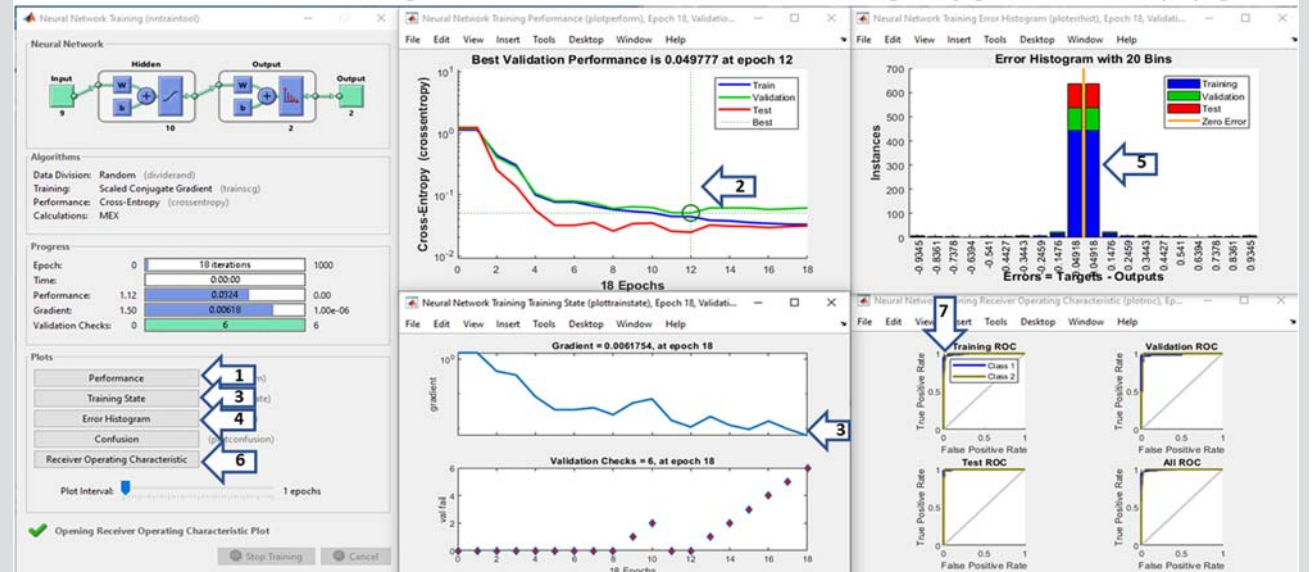
(Continued)

(Continued)

Slide 14 Description
In the MATLAB "Neural Pattern Recognition" window screen: "Neural Network Training—Conjugate Gradient Backpropagation"
In the "Train Network" section choose: click "Performance" button, its plot indicates "Best Validation Performance as 0.04977 at epoch 12."
Click "Training State" button which plot indicate a decreasing "Gradient."
Click "Error Histogram," its plots indicate very narrow error. Click "Receiver Operating Range," its plot indicates an almost perfect behavior of 90 degrees. Finally, press "Next" button in "Neural Pattern Recognition" screen

Screen figure

14. In the MATLAB "Neural Pattern Recognition" window screen: "Neural Network Training - Conjugate Gradient Backpropagation"



In the "Train Network" section choose: click "Performance" button, its plot indicates "Best Validation Performance as 0.04977 at epoch 12". Click "Training State" button which plot indicate a decreasing "Gradient". Click "Error Histogram," its plots indicate very narrow error. Click "Receiver Operating Range," its plot indicate a almost perfect behavior of 90 degree. Finally, press "Next" button in "Neural Pattern Recognition" screen.

15

In the MATLAB "Neural Pattern Recognition" window screen: "Evaluate Network"

Feel free to reevaluate the network to try to find better values with: "Train Again," "Adjust Network Size," "Import Larger Dataset," "Optionally perform additional test," etc. When ready, click the "Next" button

15. In the MATLAB "Neural Pattern Recognition" window screen: "Evaluate Network"

Neural Pattern Recognition (nprtool)

Evaluate Network

Optionally test network on more data, then decide if network performance is good enough.

Iterate for improved performance

Try training again if a first try did not generate good results or you require marginal improvement.

Train Again

Increase network size if retraining did not help.

Adjust Network Size

Not working? You may need to use a larger data set.

Import Larger Data Set

Optionally perform additional tests

Inputs: (none) ...

Targets: (none) ...

Samples are: Matrix columns Matrix rows

No inputs selected.

No targets selected.

Test Network

CE
 %E

Plot Confusion **Plot ROC**

Back **Next** **Cancel**

Select inputs and targets, click an improvement button, or click [Next].

Neural Network Start Welcome

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

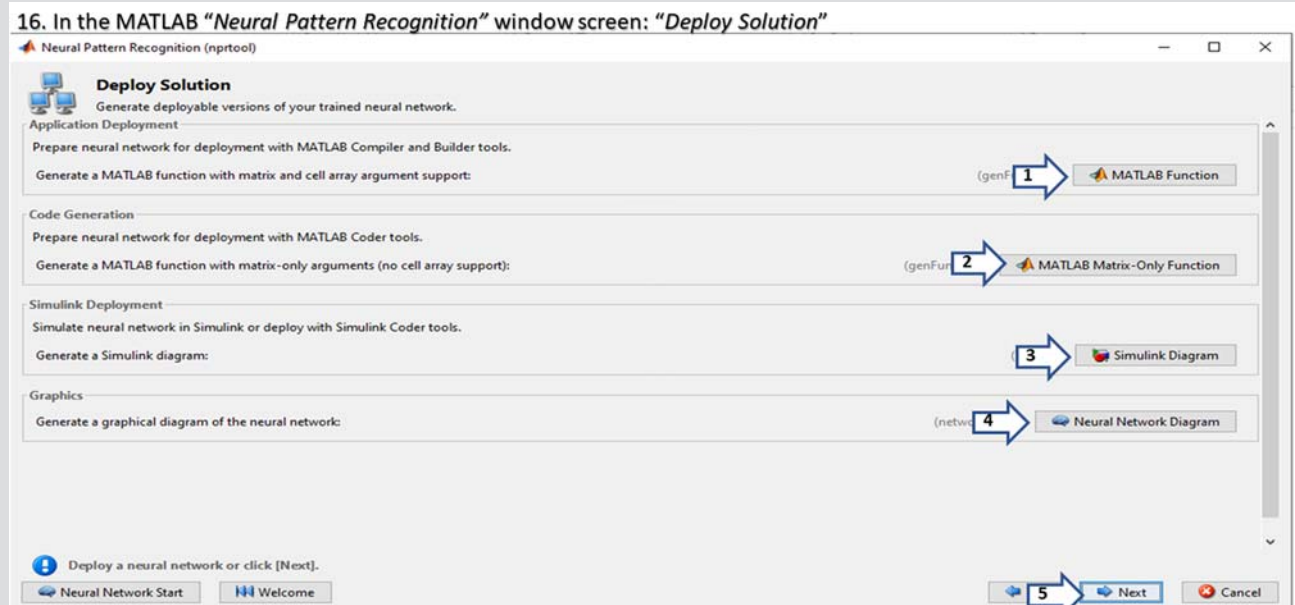
Feel free to re-evaluate the network to try to find better values with: "Train Again", "Adjust Network Size", "Import Larger Data Set", "Optionally perform additional test", etc. When ready, click the "Next" Button.

(Continued)

(Continued)

Slide 16 Description In the MATLAB "Neural Pattern Recognition" window screen: "Deploy Solution" Select the option needed for your "Deploy Solution": In the section "Application Deployment" click button "MATLAB function," In "Code Generation" click button "MATLAB Matrix-Only Function," in "Simulink Deployment" click "Simulink Diagram" and in "Graphics" click button "Neural Network Diagram." Click "Next" button

Screen figure



Select the option needed for your "Deploy Solution": In the section "Application Deployment" click button "MATLAB function", In "Code Generation" click button "MATLAB Matrix-Only Function", in "Simulink Deployment" click "Simulink Diagram" and in "Graphics" click button "Neural Network Diagram". Click "Next" Button.

17

In the MATLAB "Neural Pattern Recognition" window screen: "Save Results"

In the section "Generate Scripts" click the buttons: "Simple Script" and "Advanced Script." In the section "Save Data to Workspace" click button "Save Results." Finally click on the button "Finish," and its confirmation dialog

17. In the MATLAB "Neural Pattern Recognition" window screen: "Save Results"

Save Results
Generate MATLAB scripts, save results and generate diagrams.

Generate Scripts
Generate MATLAB scripts, save results and generate diagrams.

Recommended >> Use these scripts to reproduce results and solve similar problems.

Generate a script to train and test a neural network as you just did with this tool:

Generate a script with additional options and example code:

Save Data to Workspace

- Save network to MATLAB network object named: net
- Save performance and data set information to MATLAB struct named: info
- Save outputs to MATLAB matrix named: output
- Save errors to MATLAB matrix named: error
- Save inputs to MATLAB matrix named: input
- Save targets to MATLAB matrix named: target
- Save ALL selected values above to MATLAB struct named: results

Restore Defaults Save Results

Save results and click [Finish].

Neural Network Start Welcome Back Next Finish

In the section "Generate Scripts" click the buttons: "Simple Script" and "Advanced Script". In the section "Save Data to Workspace" click button "Save Results". Finally click on the button "Finish", and its confirmation dialog.

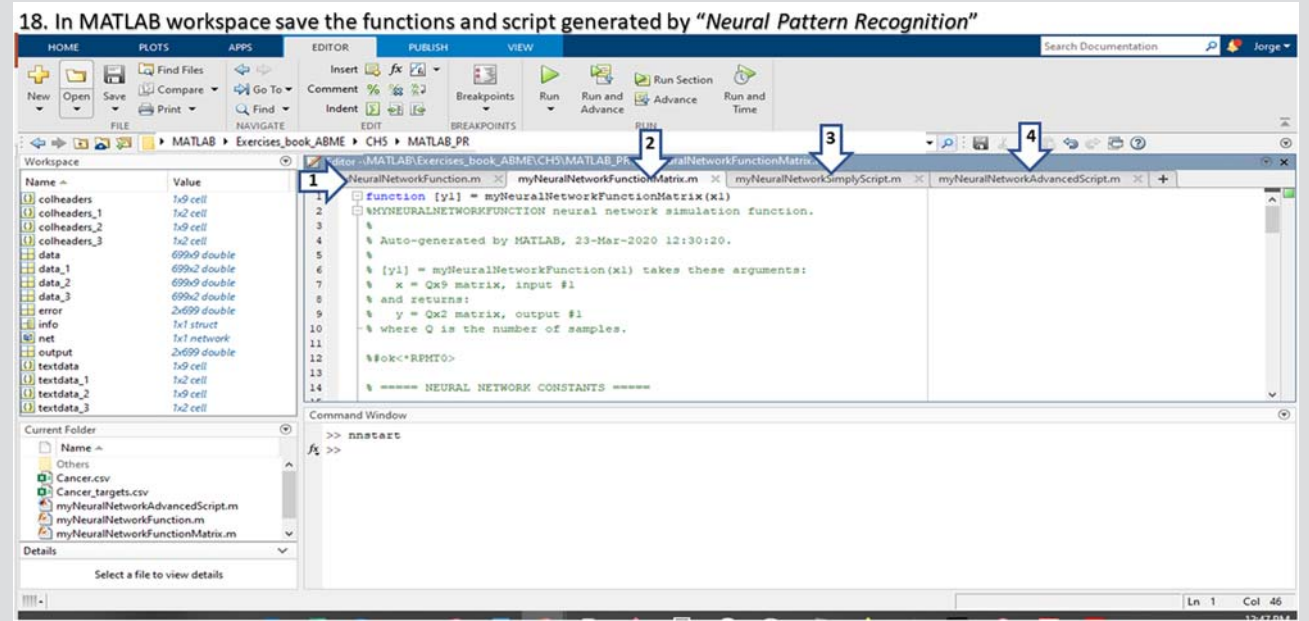
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 18 Description In MATLAB workspace save the functions and script generated by “Neural Pattern Recognition” Save the two functions generated by MATLAB “Neural Pattern Recognition” as: “myNeuralNetworkFunction.m” and “myNeuralNetworkFunctionMatrix.m.” Save also the two scripts as: “myNeuralNetworkSimplyScript.m” and “myNeuralNetworkAdvancedScript.m”

Screen figure



18. In MATLAB workspace save the functions and script generated by “Neural Pattern Recognition”

Save the two functions generated by MATLAB “Neural Pattern Recognition” as: “myNeuralNetworkFunction.m” and “myNeuralNetworkFunctionMatrix.m”. Save also the two scripts as: “myNeuralNetworkSimplyScript.m” and “myNeuralNetworkAdvancedScript.m”

19

In MATLAB workspace testing new data in "myNeuralNetworkFunctionMatrix function" generated "Neural Pattern Recognition"

To evaluate a new data as a matrix, load the file "X1.csv," show its contents with the 9 attributes for the 2 new records, call the function created by "Neural Pattern Recognition" and obtain the results for "Benign_cancer" and "Malignant_cancer," in this case the first record is a "Malignant_cancer with a 0.9885 of probability" and the second record a "Benign_cancer with 0.9965"

19. In MATLAB workspace testing new data in "myNeuralNetworkFunctionMatrix function" generated "Neural Pattern Recognition"

Workspace

Name	Value
colheaders	1x9 cell
colheaders_1	1x2 cell
colheaders_2	1x9 cell
colheaders_3	1x2 cell
data	699x9 double
data_1	699x2 double
data_2	699x9 double
data_3	699x2 double
error	2x699 double
info	1x7 struct
net	1x1 network
output	2x699 double
testdata	1x9 cell
testdata_1	1x2 cell
testdata_2	1x9 cell
testdata_3	1x2 cell

Current Folder

- Cancer.csv
- Cancer_targets.csv
- myNeuralNetworkAdvancedScript.m
- myNeuralNetworkFunction.m
- myNeuralNetworkFunctionMatrix.m
- myNeuralNetworkSimplyScript.m
- X1.csv

```
>> load('X1.csv')
>> X1
X1 =
    0.5000    0.7000    1.0000    1.0000    0.5000    1.0000    1.0000    1.0000    0.1000
    0.1000    0.1000    0.1000    0.1000    0.2000    0.1000    0.2000    0.1000    0.1000

>> [y1] = myNeuralNetworkFunction(X1)
y1 =
    0.0115    0.9885
    0.9965    0.0035
```

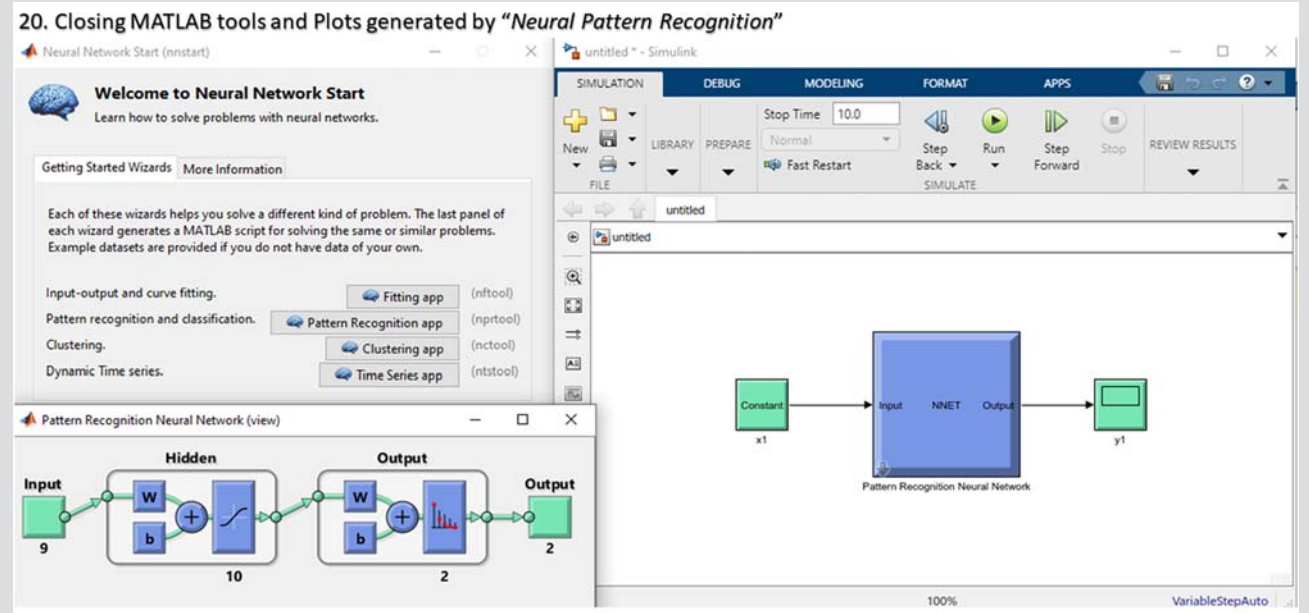
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 20 Description Closing MATLAB tools and Plots generated by "Neural Pattern Recognition" Close without saving the "Simulink diagram for the ANN" and the "graphical ANN generated" by the "Neural Network—Neural Pattern Recognition App"

Screen figure



Close without saving the "Simulink diagram for the ANN" and the "graphical ANN generated" by the "Neural Network – Neural Pattern Recognition App".

Conclusions

An AI model using MATLAB *Deep Learning Toolbox* of backpropagation Neural Network for Patterns Recognition allows the generation of a user function that represents the ANN AI model for classification of breast cancer tumors as “Benign_cancer” or “Malignant_cancer” following features of biopsies via “Fine Needle Aspiration (FNA)” in accordance with Wisconsin grade scale based on nine cytological characteristics on a scale of “1 to 10,” with “1” being the closest to benign and “10” the most anaplastic.

Recommendation

Develop Deep Learning models that are more interactive as explained in “section 6.2.5.1 Regional-CNN model for object detection of breast tumor in mammogram” to facilitate the suggestions for appropriate medical diagnosis for “breast cancer tumors*” that include: “breast tumor grading” and “breast tumor stage.”

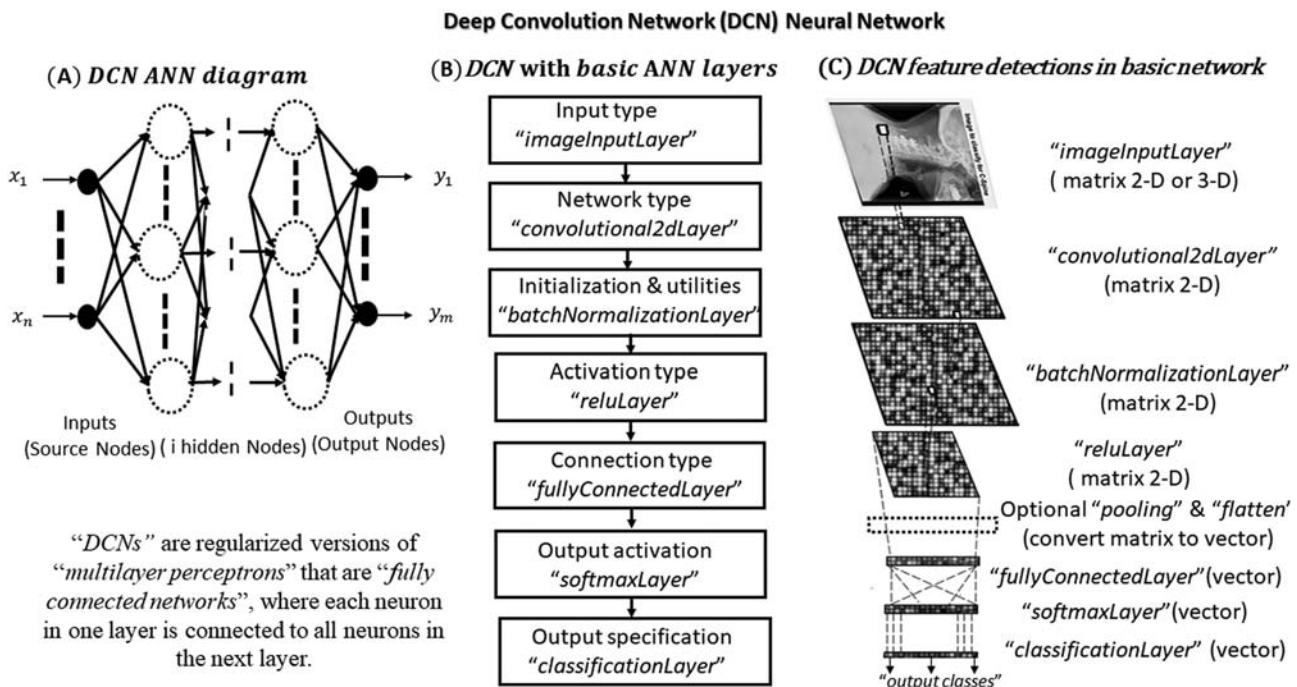
Note*: See next Research 5.5 Deep Learning using a “Pretrained Deep Convolutional Neural Network to obtain an AI model to classify Mammograms standard views types.”

5.4.5 Deep Convolution Network or ConvNet

“Deep Convolution Network (DCN) or ConvNet (CNN)” is a class of “deep neural networks,” with usually more than three hidden layers up to “n” layers (which can be hundreds) as shown in Fig. 5.14A. It combines multiple

nonlinear processing layers, using elements in parallel basically three type: “input layer,” “many hidden layers,” and an “output layer.” “DCNs” are most applied for analyzing visual imagery. They can take in an input image, assign importance based on “learnable weights” and “biases” to various aspects/objects in the image and are able to differentiate one from the other. A typical “DCN” solution architecture can be configured a list of ANN layers that transform the image volume into an output volume, these are: “input type,” “network type,” “initialization and utilities,” “activation type,” “connection type,” “output activation,” and “output specification,” as shown in Fig. 5.14B, indicating the basic layers, and Fig. 5.14C indicating their typical components:

- “Input type” can be numeric and images dataset in “2D or three-dimensional (3D),” that is, “Image Input Layer” that inputs 2D images to a network to apply “normalization.”
- “Network type” can be of many types as “Convolutional 2D or 3D layer,” and many others ANN types as, that is, “Convolutional 2D Layer” that applied sliding convolutional filter to the input.
- “Initialization and utilities” usually are determine the normalization or other type specification needed and how the information will be processed the most common is “batch” and other, as, that is, “Batch Normalization Layer” that normalize each input channel across as a mini batch.



Note: See Research 5.5 MATLAB™ Deep Learning Toolbox™ using “Deep Network Designer” to create a “custom Deep Convolutional Neural Network”

FIGURE 5.14 Deep Convolution Network (DCN) Neural Network: (A) DCN ANN diagram, (B) DCN with basic ANN layers, and (C) DCN feature detection in basic network.

- “*Activation type*” as the “*threshold where any value less than zero is set to zero,*” “*identity operation on positive inputs,*” etc. as, that is, “*Relu Layer*” that performs a threshold operation, where any value less than zero is set to zero.
- “*Connection type*” as, that is, “*fully Connected Layer*” that multiplies the inputs by a weight matrix and then add the bias vector. It delivers a vector output of “*k*” dimension, according with the number of classes*.

Note*: Some network designer recommends applying a “*pooling*” and “*flatten*” steps to make “*smooth conversion from matrix to vector.*”

- “*Output activation*”, that is, a “*Softmax Layer*” applies the “*Softmax function*” taking the input as a vector “*z*” of “*k real numbers,*” and “*normalizes*” it into a “*probability distribution consisting of k probabilities proportional to the exponentials of the input numbers.*”
- “*Output specifications*” that can be: “*classification,*” “*regression,*” “*focal loss,*” etc., that is, a “*Classification Layer*” computes the “*cross-entropy loss*” that measures the performance of a classification model whose output is a probability value between “*0*” and “*1.*” “*Cross-entropy loss increases as the predicted probability diverges from the actual label*” and can be used for multiclass classification problems with multiple exclusive classes.

The preprocessing required in a “*DCN*” is much lower compared to other classification algorithms. “*DCNs*” are regularized versions of “*multilayer perceptrons*” that are “*fully connected networks,*” where each neuron in one layer is connected to all neurons in the next layer. In the “*classical backpropagation algorithm, the weights are changed according to the gradient descent direction of an error surface*” [59]. The architecture of a “*DCN*” is analogous to that of the connectivity pattern of neurons in the human brain and was inspired by the organization of the visual cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the “*Receptive Field.*” A collection of such fields overlaps to cover the entire visual area “*DCN*” are used in application for image and video recognition, image analysis and classification, media recreation, recommendation systems, natural language processing, etc. [60,61].

Note: For an example, see: Research 5.7 MATLAB Deep Learning Toolbox using “*Deep Network Designer*” to create a “*custom Deep Convolutional Neural Network*” to obtain an AI model to “*classify Cervical X-rays view types.*”

In summary, “*DCN*” allows neural networks to improve the accuracy of image recognition classifying an input image into one of many possible classes or categories.

“*DCN*” has been applied to help in Biomedical Engineering as indicated in the following researches: Portrait style transfer using deep convolutional neural networks and facial segmentation [62], Arrhythmia Classification with ECG signals based on the Optimization-Enabled Deep Convolutional Neural Network [63], Multispecies bioacoustic classification using transfer learning of deep convolutional neural networks with pseudolabeling [64] and many more.

5.4.6 Deconvolutional network

“*Deconvolutional Network (DN)*” also known as “*deconvolutional networks (DevconvNet),*” or “*Transposed convolutional neural networks.*” “*DN*” is a neural network that performs an “*inverse convolution model*” that allows from specific class to be found inside an images, this is why it’s more frequent application is for “*object recognition inside of images,*” that is, locate tumors in body biomedical images, locate fractures in bones, etc.

“*DN*” is an unsupervised construction of hierarchical image representation learning of mid and high-level image representation, this can be achieved using feature hierarchy from alternative layers of: “*Convolutional sparse coding*” or “*Deconvolution*” that it is a method for learning by shift-invariant dictionaries in image, and “*max pooling*” that is a sample-based discretization process [35]. From the architectural point of view “*DevconvNet*” is composed of “*unpooling*” and “*deconvolution*” layers as indicated in Fig. 5.15A through Fig. 5.15D, where:

- “*Unpooling layer*” is the inverse of a “*pooling layer*” as a convolution network designed to filter from an input to a “*pooled map*” with “*switch variables: indices and size*”; meanwhile “*unpooling*” is reconstruct from the “*switch variables indices and size*” to the each maximum activation value known as “*max pooling,*” where the remembered position is used for the unpooling.
- “*Deconvolution layer**” is the conversion of the inputs from smaller size back to a larger size.

Note*: “*Deconvolutional layer*” is called also “*transposed convolutional layer.*”

A “*Deconvolution object detection net layers in basic architecture*” is shown in Fig. 5.15E, where there is a “*convolution network*” at the beginning based on series of “*maxpooling*” layers to reduce the original matrix size memorizing through using “*switch variables of indices and size*” to the each maximum activation value, and a “*deconvolution network*” at the end based on a series of “*unpooling*” layers to reconstruct the input image recognizing the object describe during the training step.

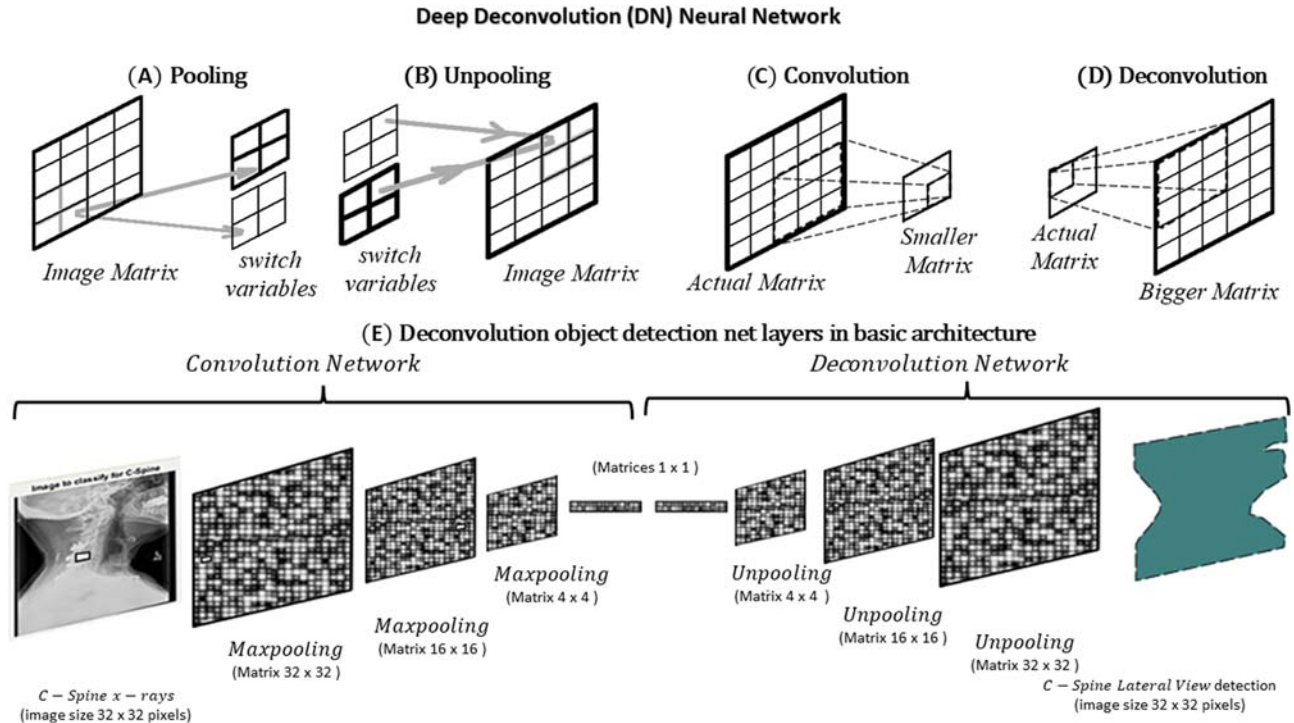


FIGURE 5.15 Deep Deconvolution (DN) Neural Network: (A) pooling versus (B) unpooling, (C) convolution versus (D) deconvolution, and (E) deconvolution object detection net layers in basic architecture.

5.4.7 Deep Convolutional Inverse Graphics Network

“Deep Convolutional Inverse Graphics Network (DCIGN)” has as objective to learn an interpretable representation of images that is disentangled with respect to various transformations such as object out-of-plane rotations, lighting variations, and texture. The “DCIGN” model is composed of multiple layers of “convolution” and “deconvolution” operators and is trained using the “Stochastic Gradient Variational Bayes (SGVB) algorithms [36]. A basic “DCIGN” uses the “CNN” as a data “encoder*” then apply the transformation needed, and finally uses a “DNN” as a “decoder*” as shown in Fig. 5.16A.

Note: Please review if necessary, the “Auto Encoder neural network” at Section 5.4.1.

Fig. 5.16B shows the “DCIGN basic architecture net layers for object detection and transformation” with a “Breast tumor X-rays” image resolution “ $N \times M$ ” as the input for the network, then the image is read as a numeric matrix and its processes the “encoder” in two steps of “convolution and pooling” layer, then the smaller matrix “ $nn \times mm$ ” is converted to various vectors where some transformation are executed to affect the image, then is converted to one vector and after that the process for “decoder” is made in two steps of “unpooling and deconvolutions” layers to complete the images with the transformation that are shown in the final output where the “breast tumor is segmented*” [65].

Note*: In Research 5.6, section 5.5.2 a “Pretrained Deep Convolutional Neural Network” is used to obtain an “AI model” to “classify Mammograms view type and suggest breast abnormalities as possible breast tumor.”

“DCIGN” is especially useful in Biomedical Engineering as shown in the following papers: NROI based feature learning for automated tumor stage classification of pulmonary lung nodules using deep convolutional neural networks [66], Multiscale brain MRI super-resolution using deep 3D convolutional networks [67], Multiplanar 3D breast segmentation in MRI via deep convolutional neural networks [68], and many others.

5.4.8 Generative Adversarial Network

“Generative Adversarial Network (GAN)” belong to the set of generative models. It means that they are able to produce/to generate new content form the input of random variables, then a generative network is trained to maximize the final classification error, with the results is generated distribution, finally a discriminate network is trained to minimize the final classification error that is used as a reference metric for both networks [69]. Besides, “Deep Convolutional Generative Adversarial Networks (DCGAN) are “GANs” which uses

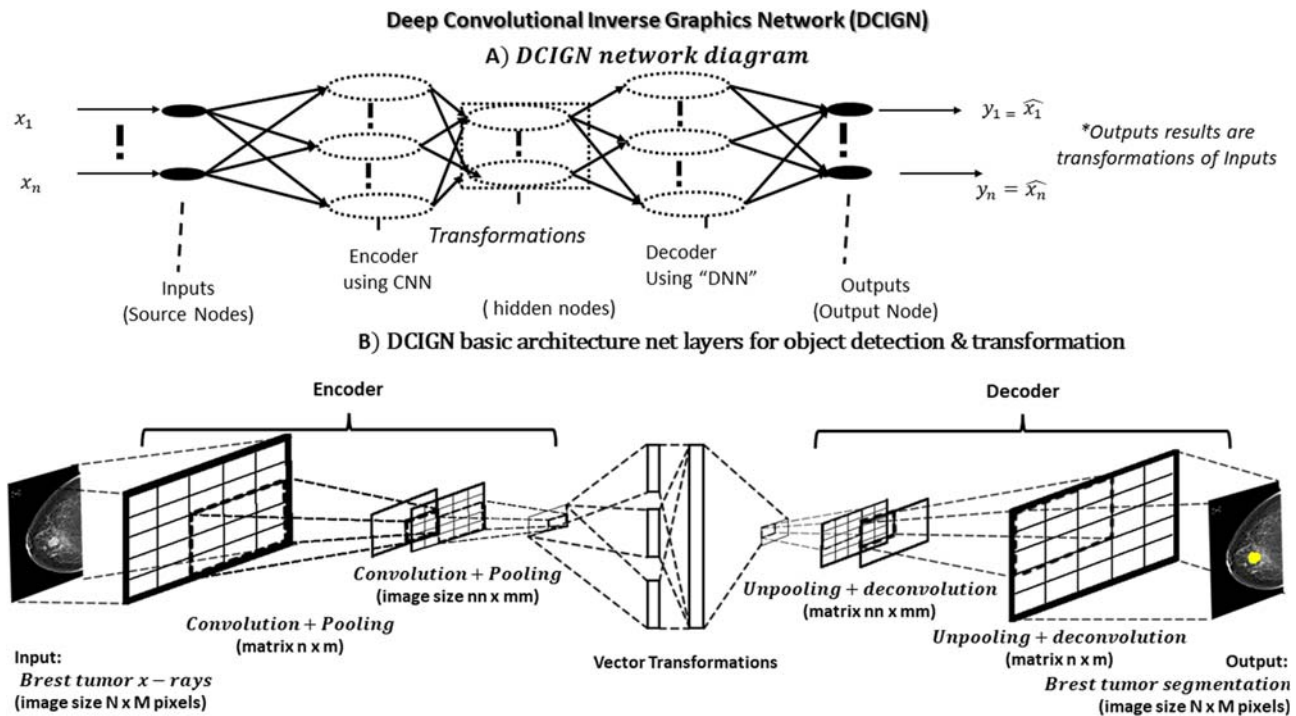


FIGURE 5.16 Deep Convolutional Inverse Graphics Network: (A) DCIGN network diagram and (B) DCIGN basic architecture net layers for object detection and transformation.

“convolutional layers.” A general “GAN” block network diagram with basically two networks blocks: “Generator” and “Discriminator” as shown in Fig. 5.17A, where:

- “Generator” is a network that has inputs as a vector of random values known as “latent inputs,” it has the task of generate data with the same structure as the “training data” as shown in Fig. 5.17B.
- “Discriminator” is a network that has batches of data containing observations from both the “training data” and “generated data” from the generator as shown in Fig. 5.17C. The “discriminator” attempts to classify the observations as “real or generated” using the “conditional probability” shown in Eq. (5.2).

$$\text{GAN Classifier conditional probability } P(X|Y) \quad (5.2)$$

where X is the input and Y is the label.

The training of a “GAN” is made in both networks simultaneously to maximize the performance of both, in the “generator” the training is for generate data to “fool” the discriminator and in the “discriminator” the training has the objective to distinguish between “real and generated data.”

The “generative network” is trained to maximize the final classification error between “true and generated data,” while the “discriminative network” is trained to minimize it. This is where the notion of adversarial networks arises from.

“GANs’ are has been applied to obtain different AI models in Biomedical Engineering as: PCSGAN: Perceptual cyclic-synthesized generative adversarial networks for thermal and near infrared to visible image transformation [70], Skin lesion segmentation via generative adversarial networks with dual discriminators [71], Plausibility-promoting generative adversarial network for abstractive text summarization with multitask constraint [72], and many others.

5.4.9 Deep Residual Network or Deep ResNet

“Deep Residual Network (DRN) or Deep ResNet” is a neural network using a “residual learning framework” that preserve inputs and improve accuracy using layers typically up to 150 or more, instead of trying the solution mapping some input to same output in all the layers usually, the network try to learn to map some input to some output, this mean that has an “X identity solution” carrying old inputs over and serving it freshly to a later layer, this is made using a “residual learning block” as shown in Fig. 5.18A, that is repeated in all the layer network as indicated in Fig. 5.18B.

Deep ResNet architecture holds many staked layers including “convolutional,” “pooling,” and “fully connected.” This network has a “skip function” that reduces the number of times a linear function is used to achieve

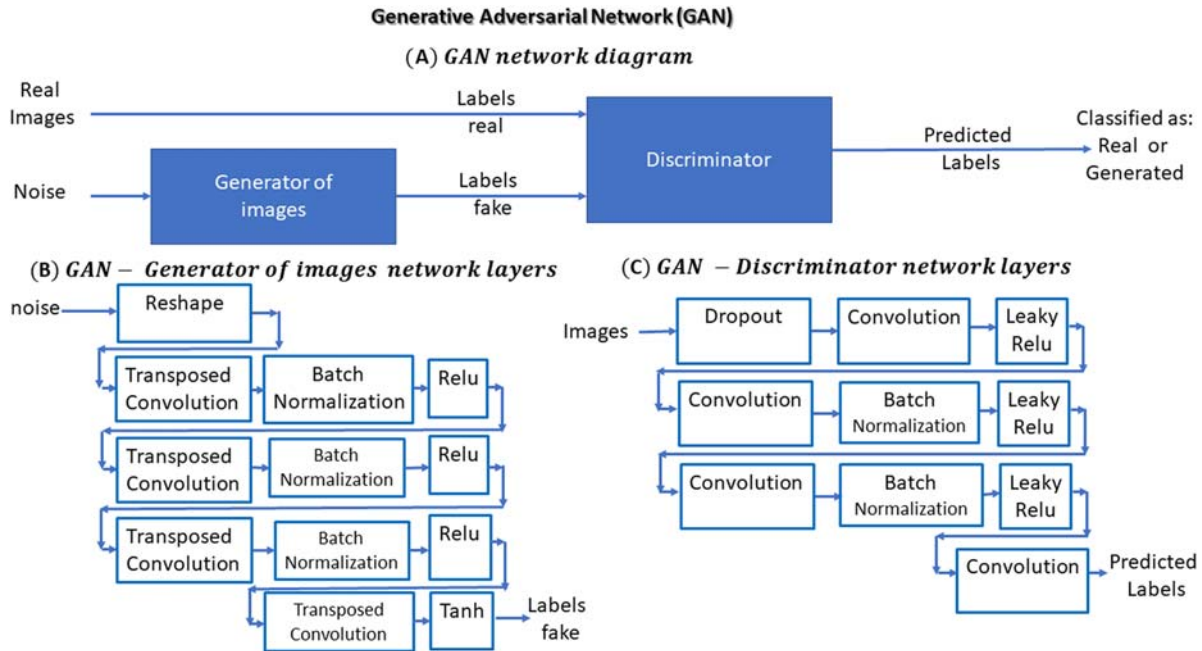


FIGURE 5.17 Generative Adversarial Network (GAN): (A) GAN neural network diagram, (B) GAN-Generator of images network layers, and (C) GAN-discriminator network layers.

Deep Residual Network (DRN) or Deep ResNet

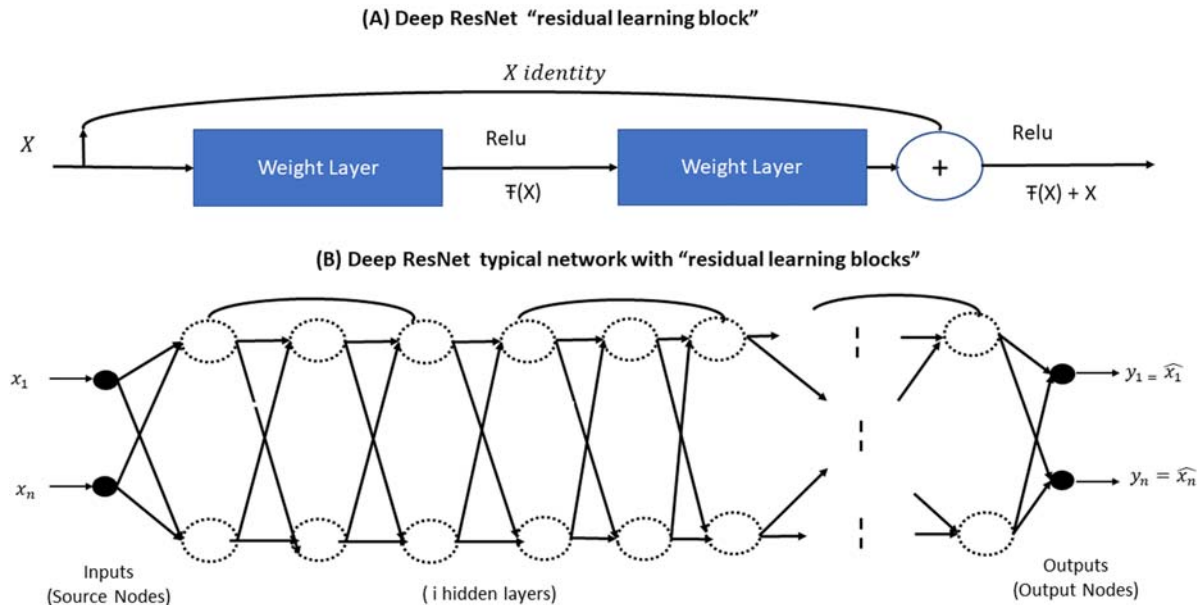


FIGURE 5.18 Deep Residual Network (DRN) or Deep ResNet: (A) Deep ResNet “residual learning block” and (B) Deep ResNet typical network with “residual learning blocks.”

an output creating “residual block” that eliminate the “varnished gradients,” this mean when a gradient becomes very small, that even a very big change in the input will not affect the output as desired, and “exploding

gradients” when a gradient becomes exponentially big, that the algorithm can no longer be used to train the model. “DRN are very used in Biomedical for image recognition applications.”

It has been proven that “DRN” networks are in essence just “Recurrent Neural Network (RNNs*) without the explicit time based construction” and they are often compared to “Long/Short-Term Memory (LSTM),” “Gated Recurrent Unit (GRU),” and “Recurrent Neural Network (RNN) without gates” [73].

Other “DRN” applications in Biomedical Engineering are: An efficient brain tumor image segmentation based on deep residual networks (ResNets) [74], Towards efficient medical lesion image super-resolution based on deep residual networks [75], A deep Residual U-Net convolutional neural network for automated lung segmentation in computed tomography images [76], and many others.

Note*: “Recurrent Neural Network (RNN)” are going to be studied in the next chapter with different researches as examples at section 6.2

5.5 Transfer learning from pretrained deep learning networks

As we can see “ANN” in “Deep Learning” can be extraordinarily complex and time-consuming to design, build, train, and test from the ground up the neural network for some practical “AI applications.” Fortunately, each time there are more and more “Pretrained Deep Learning Networks” developed to handle large dataset that allow to generate easily “AI models” for practical applications. These kind of “pretrained deep learning networks” are very fast because, they are designed based on powerful

and complex new “System Architectures” that are discovered ever year using powerful computational power based on very fast and expensive computer hardware, designed with the special purpose of handle general images that can be used in many “AI applications”: “self-driving vehicles,” “detecting abnormalities in medical imaging,” and many more.

The “pretrained deep learning networks” are basically “AI image classifiers” that allow “AI image feature extraction,” “prediction,” and others special “AI features” as “Transfer Learning” for endless “AI applications.” Where, “Transfer Learning” consists in taking layers from a pretrained network on large dataset and fine-tune on for new custom dataset, in this book focus for “Biomedical Engineering applications.” Some examples “Pretrained Deep Learning Networks” available for public applications are summarized in Fig. 5.19.

Usually the “Pretrained Deep Learning Networks” can be imported and exported using the “Open Neural Network Exchange (ONNX) model format,” but also from “TensorFlow-Keras,” “Caffe,” “PyTorch,” “MxNet,” “Core ML,” “Microsoft Cognitive Toolkit,” “MATLAB,” IBM Watson and others.

A set of research on “Biomedical Engineering” are shown in the next three tutorial examples to be familiarize with “pretrained deep convolutional” and to design, build, training and deploy “custom AI models”:

- Research 5.5, section 5.5.1 MATLAB Deep Learning Toolbox using a “Pretrained Deep Convolutional

Some “Pretrained Deep Learning Networks”

Pretrained DNN	Description	Layers Deep /Categories	Image input Size	Millions of parameters
googlenet	“Convolutional neural network” trained on “ImageNet” classifies images into 1000 object categories, such as keyboard, mouse, pencil, and many animals	22 / 1000	224-by-224	1.24
resnet101	“Residual learning framework” to ease the training of networks that are substantially deeper. Allows reformulate layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions	101 / 1000	224-by-224	44.6
inceptionresnetv2	“Convolutional neural network” trained on more than a million images from the ImageNet database	164 / 1000	299-by-299	55.9
densenet201	“Dense Convolutional Network” identity mapping is proposed to promote the gradient propagation. Element-wise addition is used. It can be viewed as algorithms with a state passed from one ResNet module to another one.	201 / 1000	244-by-224	20
nasnetlarge	“Large convolutional Network” trained on more than a million images from the ImageNet database	No linear layers / 1000	331-by-331	88.9
vgg19	“Very Deep Convolutional Networks” for Large-Scale Image Recognition increasing depth using an architecture with very small (3x3) convolution filters	19 / 1000	224-by-224	144

FIGURE 5.19 Some “Pretrained Deep Learning Networks” that simplify the generation of “AI models.”

Neural Network to obtain an AI model to classify Mammograms standard views types.”

- Research 5.6, section 5.5.2 MATLAB Deep Learning Toolbox using “Deep Network Designer” to modify a “Pretrained Deep Convolutional Neural Network” to obtain an AI model to “classify Mammograms view type and suggest breast abnormalities as possible breast tumor.”
- Research 5.7, section 5.5.3 MATLAB Deep Learning Toolbox using “Deep Network Designer” to create a “custom Deep Convolutional Neural Network” to obtain an AI model to “classify Cervical X-rays view types.”

The main goal of the next three researches as examples are to prove the advantages of using “Pretrained Deep Learning Networks” or develop “custom AI networks” for endless number of applications in “Biomedical Engineering” to obtain faster “AI models” for a big diversity of problems to resolve.

5.5.1 Research 5.5 “Pretrained Deep Convolutional Neural Network to obtain an AI model to classify Mammograms standard views types”

5.5.1.1 Case for research

Obtain an AI model using a modified “Deep Convolutional Neural Network (DCN)” based on a “Modified Pretrained DCN as GoogLeNet to Classify Mammograms standard views types.”

5.5.1.2 General objective

Apply modification to a “Pretrained DCN as GoogLeNet using MATLAB Deep Learning Toolbox, replacing layers of the same GoogLeNet to create new categories or classes to obtain an AI Model to Classify Mammograms standard views types.”

5.5.1.3 Specific objectives

- Load the “Pretrained DCN GoogLeNet” with its original 1000 categories.
- Test the original “GoogLeNet” with a “mammography images” to classifying using the original categories “GoogLeNet” to obtain its “Top 5 Predictions.”
- Create “mammograms categories from a images dataset” defining a “MATLAB Image Datastore.”
- Analyze the structure of the “DCN GoogLeNet” with its 144 original layers and its standard images size.
- Locate the final “GoogLeNet” layers and replace two of its final layers with the same type for the purpose of “mammography images.”

- Frozen the first 10 initial layers of “GoogLeNet” to save processing time in the classification model.
- Specify the training options according with the hardware available.
- Train the “modified GoogLeNet” to obtain an “AI DCN model for classification of mammograms.”
- “Classify Validation images and show random images with score” using the “AI DCN model for classification of mammograms standard view types.”
- “Classify a mammography images and obtain its score using the mammography standard view types” applying the “AI DCN model for classification of mammograms views types.”
- Obtain the “4 Top predictions and calculate its class probabilities.”
- Deploy and test the AI model obtained with mammography to classification with score of the “top 4 predictions for Mammograms standard views types.”

5.5.1.4 Background for “Mammography views”

There are two techniques for creating a “mammogram” that used a “low-doses X-rays” machine: “Film-screen mammography” that creates a photographic film, and “digital mammography” that creates digital images. Both methods use the same procedure for taking the image: the person having the mammogram will place their breast between two clear plates, which will squeeze it between them to hold it in place. This flattens the breast for a better image and stops the image from blurring. The machine takes a picture of the breast from two angles, each are known “mammography view types.” The images can be split in two groups: “standard views” and “supplementary views,” where [77]:

- “Standard views” are four views identified as: two of “45-degree Medio Lateral Oblique (MLO)” and two of “Craniocaudal View (CC)” as shown in Fig. 5.20, where:
 - “MLO Mammography view type” is taken in both breasts: right breast identified as “RMLO” and left breast as “LMLO.” Because the “Mammogram views” is a 2D representation of a 3D structure, the “MLO view is taken with 45 degrees” extra tissue is taken without extra exposure.
 - “CC Mammography view type” is taken in both breasts: right breast identified as “RCC” and left breast as “LCC.”

Note: Not all 4 views are always performed in all “mammogram” studies. For instance, in clients under 40 years old, only 2 MLO views may be done to limit radiation of “low dose X-rays” exposure, depending on local policy and the discretion of the radiologist. Other exceptions are in cases of recent surgery or when the

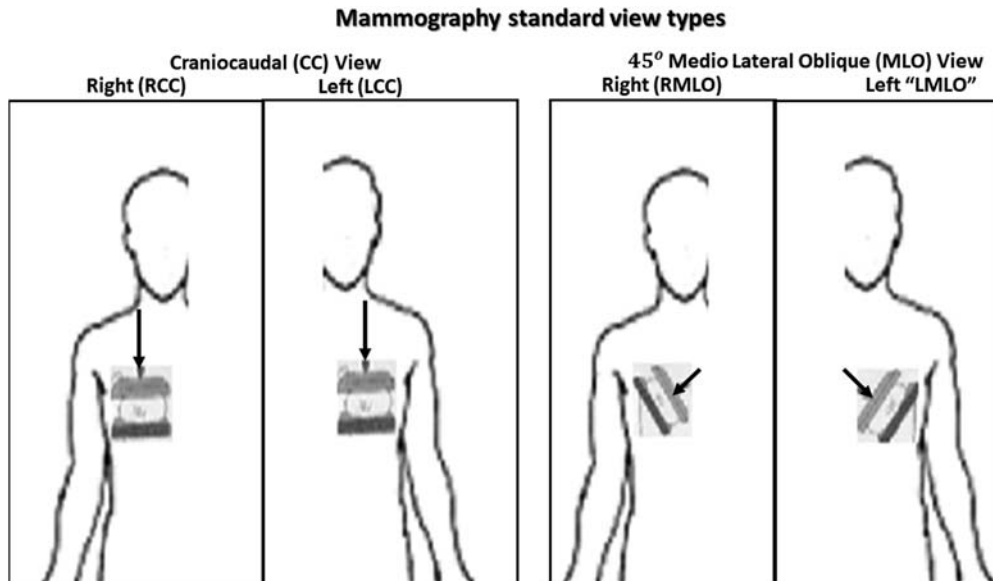


FIGURE 5.20 Mammography standard view types: left and right craniocaudal (CC) views, and left and right 45-degree mediolateral oblique (MLO) views.

patient has painful breast pathology or large lesions or an abscess.

- “Supplementary views” are only taken when it is necessary to detect additional information as:
 - “True Lateral view-90 degrees” identified as: “Mediolateral (ML)” and “Lateromedial (LM).”
 - “Exaggerated craniocaudal views” identified as: “XCCL” and “XCCM.”
 - “Magnification views,” and other views.

Note: Additional views are frequently followed by ultrasound if there is a positive finding, because the additional ultrasound views add more complementary information.

5.5.1.5 Dataset

The “low-dose X-ray” image dataset from screening “Mammography” is organized in four folders that contains

the images from different patients with a “*image size known as pixel resolution of 2048 × 2432.*” The folder name defines the mammography category with the following standard views name: “Breast LCC,” “Breast LMLO,” “Breast RCC,” and “Breast RMLO.” The folder organization, and their respective images are shown in Fig. 5.21.

5.5.1.6 Procedure

The steps to obtain a “AI Convolutional Neural Network (CNN) model” using a “Pretrained Deep Convolutional Neural Network to obtain an AI model to classify Mammograms standard views types” are summarized in Table of slides 5.5 and each step of the example are visually explained using screen sequences with instructions in easy to follow screen figures.

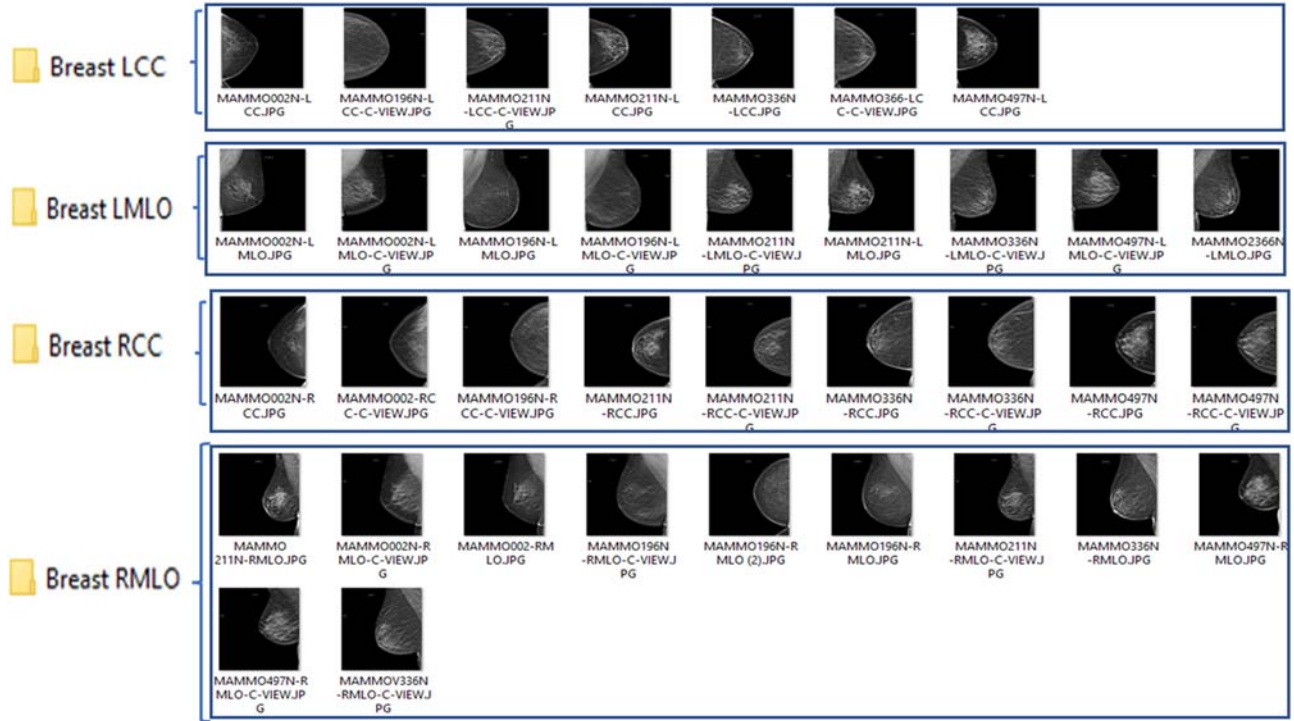


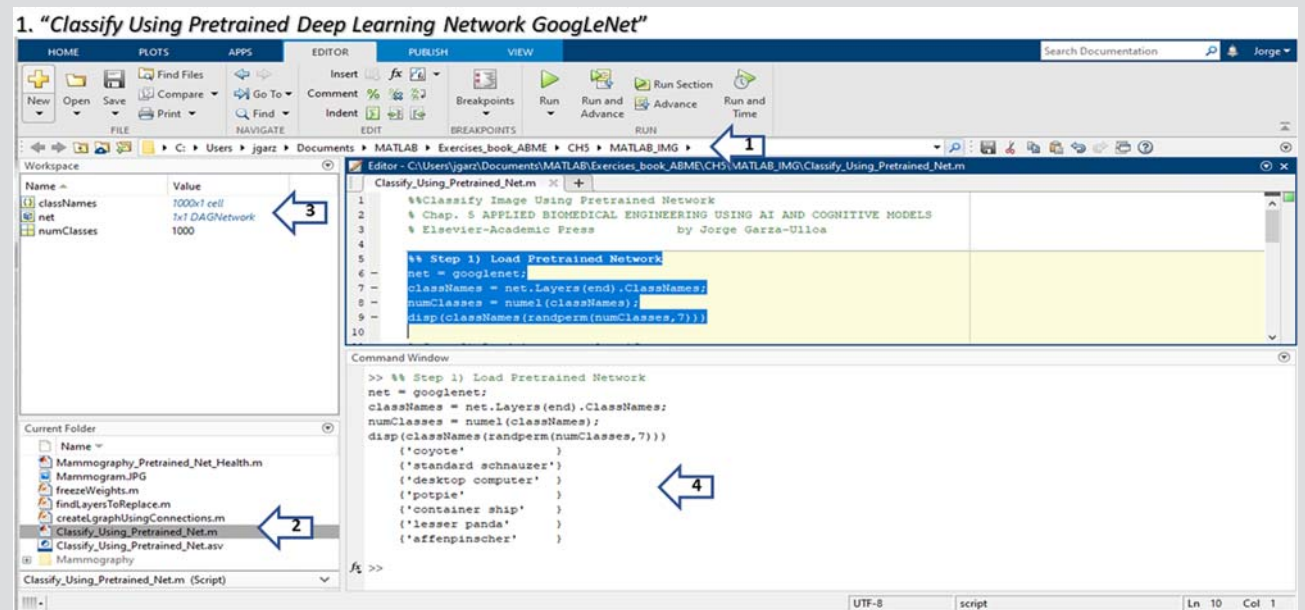
FIGURE 5.21 Image Dataset “*Mammography*” is organized in for four folders according with the “*Mammograms standard views type.*” Note: This image dataset is available in the companion directory of the book, in the following directory: “. . .\Exercises_book_ABME\CH5\MATLAB_IMG\Mammography.”

Table of slides 5.5 MATLAB Deep Learning Toolbox using a “Pretrained Deep Convolutional Neural Network to obtain an AI model to classify Mammograms.”

Slide Description

Screen figure

1 *Classify Using Pretrained Deep Learning Network: GoLeNet in MATLAB*
 Open MATLAB software and select the directory “...\\Exercices_book_ABME\\CH5\\MATLAB_IMG.” Open the MATLAB script “Classify_Using_Pretrained_Net.m,” copy on the command prompt the step 1), where: “GoLeNet” is loaded to the variable ‘net’. It has ‘numClasses = 1000’, and their respective “classNames.” This step list randomly 5 of the 1000 images categories available

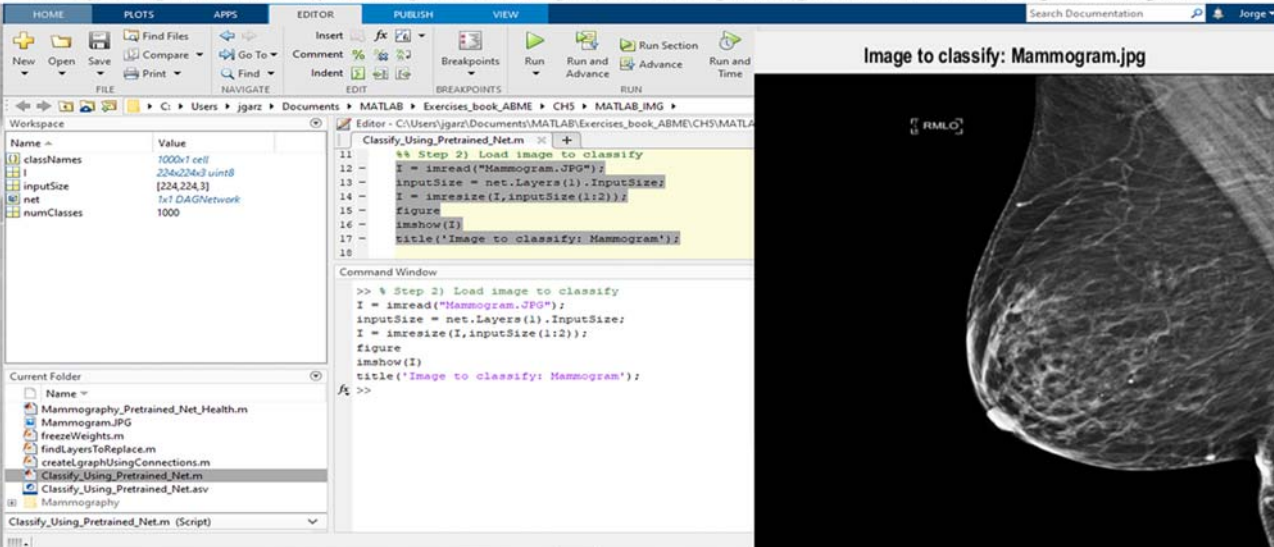


Open MATLAB software & select the directory “...\\Exercices_book_ABME\\CH5\\MATLAB_IMG”. Open the MATLAB script “Classify_Using_Pretrained_Net.m”, copy on the command prompt the step 1), where: “GoLeNet” is loaded to the variable ‘net’. It has ‘numClasses=1000’, and their respective “classNames”. This step list randomly 5 of the 1000 images categories available
 From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

2

“Classify Using Pretrained Deep Learning Network GoogLeNet”: Loading an image for classification of original “GoogLeNet Categories”
Copy and paste on the command prompt the step 2) of the MATLAB script, where: “Mammogram.JPG” is loaded, resized from its original X-rays resolution of 2048×2432 pixels to the “GoogLeNet” resolution of “ 224×224 ,” then the image is shown

2. “Classify Using Pretrained Deep Learning Network GoogLeNet”: Loading an image for classification of original “GoogLeNet



The screenshot shows the MATLAB environment. The script editor contains the following code:

```
11 % Step 2) Load image to classify
12 I = imread("Mammogram.JPG");
13 inputSize = net.Layers(1).InputSize;
14 I = imresize(I,inputSize(1:2));
15 figure
16 imshow(I)
17 title('Image to classify: Mammogram');
18
```

The Command Window shows the execution of the script:

```
>> % Step 2) Load image to classify
I = imread("Mammogram.JPG");
inputSize = net.Layers(1).InputSize;
I = imresize(I,inputSize(1:2));
figure
imshow(I)
title('Image to classify: Mammogram');
fx >>
```

The figure window displays the loaded mammogram image titled "Image to classify: Mammogram.jpg".

Copy and paste on the command prompt the step 2) of the MATLAB script, where : "Mammogram.JPG" is loaded, resized from its original x-rays resolution of 2048×2432 pixels to the “GoogLeNet” resolution of “ 224×224 ”, then a figure is generated as shown.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

3
"Classify Using Pretrained Deep Learning Network GoogLeNet":
"Classifying with the available GoogleNet classes"
Copy and paste on the command prompt the step 3) of the MATLAB script. Where, the AI Classification of the image is compared with the images of the 1000 categories available, and its best classification is a "nematode with 28.2% of probability"

3. "Classify Using Pretrained Deep Learning Network GoogLeNet": "Classifying with the available GoogleNet classes"

The screenshot displays the MATLAB environment. The Command Window shows the execution of the following MATLAB code:

```
>> [label, scores] = classify(net, I);  
figure  
imshow(I)  
title('GoogleNet MAMMOGRAPHY = '+string(label) + ', ' + ...  
num2str(100*scores(classNames == label),3) + '%');  
disp(classNames(label))  
'nematode'
```

The Workspace window shows the following variables:

Name	Value
classNames	1000x7 cell
I	224x224x3 uint8
inputSize	[224,224,3]
label	1x1 categorical
net	1x1 DAGNetwork
numClasses	1000
scores	1x1000 single

The Current Folder window shows the following files:

- Mammography_Pretrained_Net_Health.m
- Mammogram.JPG
- freezeWeights.m
- findLayersToReplace.m
- createGraphUsingConnections.m
- Classify_Using_Pretrained_Net.m
- Classify_Using_Pretrained_Net.asv
- Mammography
- Classify_Using_Pretrained_Net.m (Script)

The resulting image is a mammogram with a title "GoogleNet MAMMOGRAPHY = nematode, 28.2%" and a blue arrow pointing to a region of interest.

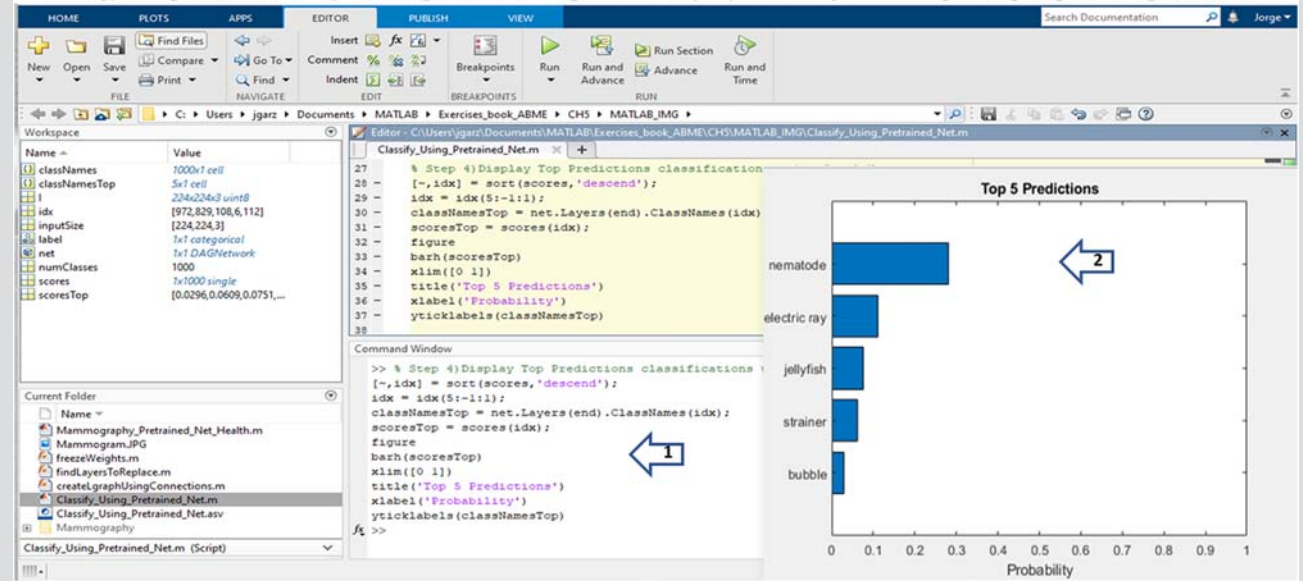
Copy and paste on the command prompt the step 3) of the MATLAB script. Where, the AI Classification of the image is compared with the images of the 1000 categories available, and its best classification is a "nematode with 28.2% of probability".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

4

“Classify Using Pretrained Deep Learning Network GoogLeNet”: “Top 5 prediction for the image using original GoogLeNet classes”
Copy and paste on the command prompt the step 4) of the MATLAB script. Where a chart is generated indicating the *“Top 5 Predictions using the original GoogLeNet categories.”* Finally, close the script from the script *“Classify_Using_Pretrained_Net.m”*

4. *“Classify Using Pretrained Deep Learning Network GoogleNet”: “Top 5 prediction for the image using original GoogleNet classes”*



Copy and paste on the command prompt the step 4) of the MATLAB script. Where a chart is generated indicating the *“Top 5 Predictions using the original GoogleNet categories.”* Finally, close the script from the script *“Classify_Using_Pretrained_Net.m”*

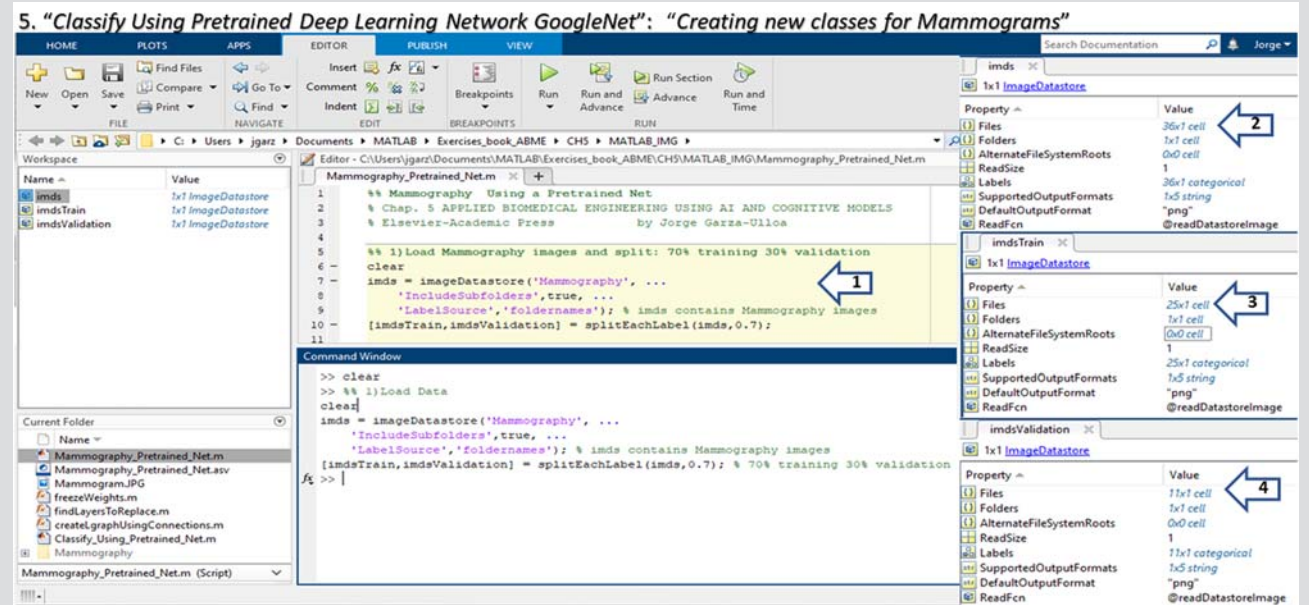
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

- Slide 5 Description
- “Classify Using Pretrained Deep Learning Network GoogleNet”: “Creating new classes for Mammograms”
- Open the MATLAB script “Mammography_Pretrained_Net.m”, copy on the command prompt the step 1), that clear all and create the variables: “imds with an image Datastore of 36 mammograms images,” “imdsTrain with 25 of them (70%)” and “imdsValidation with 11 of them (30%)”

Screen figure



Open the MATLAB script “Mammography_Pretrained_Net.m”, copy on the command prompt the step 1), that clear all and create the variables: “imds with an image Datastore of 36 mammograms images”, “imdsTrain with 25 of them (70%)” and “imdsValidation with 11 of them (30%)”

6

“Classify Using Pretrained Deep Learning Network GoogLeNet”: “Load Pretrained Network and Analyze Network”

Copy and paste on command prompt the step 2) where: “GoogLeNet” is loaded in variable ‘net’, the command “analyzeNetwork” generates a “Deep Learning ANN map with 144 layers,” the beginning and the end are shown at the left. The original “GoogLeNet” standard images size are “244 × 244 pixels × 3 basic colors”

6 “Classify Using Pretrained Deep Learning Network GoogLeNet”: “Load Pretrained Network & Analyze Network”

The screenshot displays the MATLAB environment with the following components:

- Workspace:** Lists variables such as `ans`, `imsds`, `imsdsTrain`, `imsdsValidation`, `inputSize`, and `net`.
- Script Editor:** Contains MATLAB code for loading the pretrained network and analyzing it. Key lines include:


```
net = googlenet; % Load Pretrained Network
analyzeNetwork(net) % Generate a Network diagram
net.Layers(1) % Analyze layer 1
inputSize = net.Layers(1).InputSize % Find inputSize
```
- Command Window:** Shows the execution of the script, including the command `analyzeNetwork` and the resulting network diagram.
- Network Diagram:** A visual representation of the 144-layer GoogLeNet architecture, showing various layers like convolution, pooling, and fully connected layers.
- Analysis Results Table:** A table providing detailed information about each layer in the network, including its name, type, activation function, learnability, and input/output sizes.

Copy and paste on command prompt the step 2) where: “GoogLeNet” is loaded in variable ‘net’, the command “analyzeNetwork” generates a “Deep Learning ANN map with 144 layers”, the beginning and the end are shown at the left. The original “GoogLeNet” standard images size are “244 x 244 pixels x 3 basic colors”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

7 “Classify Using Pretrained Deep Learning Network GoogLeNet”: “Find the Final Layers of GoogleNet”
Copy and paste on command prompt the step 3a) where: variable ‘lgraph’ is loaded with all 144 original layers, then the two layers to be replaced must have the types and properties as specified, these are found using the user function “FindLayersToReplace”

7. “Classify Using Pretrained Deep Learning Network GoogleNet”: “Find the Final Layers of GoogleNet”

```
18 %% 3) Replace Final two Layers with the same type
19 %3a) Load lgraph=Layer graphs with properties
20 if isa(net, 'SeriesNetwork')
21     lgraph = layerGraph(net.Layers);
22 else
23     lgraph = layerGraph(net);
24 end
25 lgraph % Display lgraph
26 % Find names of two layer to replace
27 [learnableLayer,classLayer] = findLayersToReplace(lgraph);
28 [learnableLayer,classLayer] % Show the name of the two final layers
```

Command Window

LayerGraph with properties:

Layers: [144x1 nnet.cnn.layer.Layer]
Connections: [170x2 table]
InputNames: {'data'}
OutputNames: {'output'}

ans =

1x2 Layer array with layers:

1	'loss3-classifier'	Fully Connected	1000 fully connected layer
2	'output'	Classification Output	crossentropyex with 'tench' and 999 other classes

Copy and paste on command prompt the step 3a) where: variable ‘lgraph’ is loaded with all 144 original layers, then the two layers to be replaced must have the types and properties as specified, these are found using the user function “FindLayersToReplace”

8

“Classify Using Pretrained Deep Learning Network GoogLeNet”:
“Replace two of the Final Layers with the same type”
Copy and paste on command prompt the step 3b) where: the two of the last layers are replaced as “new_fc” and “new_classoutput”

8. “Classify Using Pretrained Deep Learning Network GoogLeNet”: “Replace two of the Final Layers with the same type”

The screenshot shows the MATLAB Editor interface. The main window displays the code for the `Mammography_Pretrained_Net.m` script. The code includes comments and logic to define new layers and replace existing ones in the network graph. Three blue arrows point to specific parts of the code: arrow 1 points to the `newLearnableLayer = fullyConnectedLayer` line, arrow 2 points to the `Name: 'new_fc'` property, and arrow 3 points to the `OutputSize: 4` property. The Command Window at the bottom shows the properties of the `newLearnableLayer`, confirming it is a `FullyConnectedLayer` with the specified name and output size.

```
29 % 3b) Define new layers and replace them
30 numClasses = numel(categories(imdsTrain.Labels));
31 if isa(learnableLayer, 'nnet.cnn.layer.FullyConnectedLayer') % Definelayers type
32 newLearnableLayer = fullyConnectedLayer(numClasses, ...
33     'Name', 'new_fc', ...
34     'WeightLearnRateFactor', 10, ...
35     'BiasLearnRateFactor', 10);
36 elseif isa(learnableLayer, 'nnet.cnn.layer.Convolution2DLayer')
37 newLearnableLayer = convolution2dLayer(1, numClasses, ...
38     'Name', 'new_conv', ...
39     'WeightLearnRateFactor', 10, ...
40     'BiasLearnRateFactor', 10);
41 end
42 newLearnableLayer % display type of layer to replace
43 lgraph = replaceLayer(lgraph, learnableLayer.Name, newLearnableLayer); % Replace Layer
44 newClassLayer = classificationLayer('Name', 'new_classoutput');
45 lgraph = replaceLayer(lgraph, classLayer.Name, newClassLayer);
```

Command Window
newLearnableLayer =
FullyConnectedLayer with properties:
Name: 'new_fc'
Hyperparameters
InputSize: 'auto'
OutputSize: 4
Learnable Parameters
Weights: []
Bias: []
[Show all properties](#)

Copy and paste on command prompt the step 3b) where: the two of the last layers are replaced as “new_fc” and “new_classoutput”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

9 “Classify Using Pretrained Deep Learning Network GoogLeNet”: “Verifying the final two Layers were replaced with the same type”
Copy and paste on command prompt the step 3c) where: a “lgraph ANN map” is plotted and observe the last two layers replaced with the same type of layers

9. “Classify Using Pretrained Deep Learning Network GoogLeNet”: “Verifying the final Layers were replaced with the same type”

Workspace

Name	Value
ans	1x2 Layer
classLayer	1x1 ClassificationOut
imds	1x1 ImageDatastore
imdsTrain	1x1 ImageDatastore
imdsValidation	1x1 ImageDatastore
inputSize	[224,224,3]
learnableLayer	1x1 FullyConnectedL
lgraph	1x1 LayerGraph
net	1x1 DAGNetwork
newClassLayer	1x1 ClassificationOut
newLearnableLayer	1x1 FullyConnectedL
numClasses	4

Current Folder

- Mammography_Pretrained_Net.m
- Mammography_Pretrained_Net.asv
- Mammogram.JPG
- freezeWeights.m
- findLayersToReplace.m
- createLgraphUsingConnections.m
- Classify_Using_Pretrained_Deep_Learning_Net.m
- findLayersToReplace.m (Function)

```
46 %3c) Verify the layers replaced
47 - figure('Units','normalized','Position',[0.3 0.3 0.4
48 - plot(lgraph) % Show the last part of lgraph map
49 - ylim([0,10])
50
```

```
>> %3c) Verify the layers replaced
figure('Units','normalized','Position',[0.3 0.3 0.4]);
plot(lgraph)
ylim([0,10])
>>
```

Copy and paste on command prompt the step 3c) where: a “lgraph ANN map” is plotted and observe the last two layers replaced with the same type of layers.

10

“Classify Using Pretrained Deep Learning Network GoogLeNet”:

“Network with initial layers frozen to save time”

Copy and paste on command prompt the step 4a) where: the 10 first layers are frozen to save classification processing time in this net already pretrained, create a “imageAugmenter,” and automatically resize training images for: training “augimdsTrain” and “augiValidation” to “244 × 244 pixels”

10. “Classify Using Pretrained Deep Learning Network GoogleNet”: “Network with initial layers frozen to save time”

The screenshot shows the MATLAB R2020a environment with the following components:

- Workspace:** Lists variables such as `ans`, `augimdsTrain`, `augimdsValidation`, `classLayer`, `connections`, `imageAugmenter`, `imds`, `imdsTrain`, `imdsValidation`, `inputSize`, `layers`, `learnableLayer`, `lgraph`, `net`, `newClassLayer`, `newLearnableLayer`, and `numClasses`.
- Current Folder:** Shows files like `Mammography_Pretrained_Net.asv`, `Mammogram.JPG`, `freezeWeights.m`, `findLayersToReplace.m`, `createLgraphUsingConnections.m`, and `Classify_Using_Pretrained_Net.m`.
- Editor:** Contains the script `Classify_Using_Pretrained_Net.m` with the following code:

```
51 % 4) Train Network with initial layers frozen to save time
52 % 4a) Freeze the first 10 layers to save processing time
53 layers = lgraph.Layers;
54 connections = lgraph.Connections;
55 layers(1:10) = freezeWeights(layers(1:10));
56 lgraph = createLgraphUsingConnections(layers,connections);
57 % Use an augmented image
58 imageAugmenter = imageDataAugmenter;
59 augimdsTrain = augmentedImageDatastore(inputSize(1:2),imdsTrain, ...
60 'DataAugmentation',imageAugmenter);
61 % Automatically resize training images without any additional preprocessing
62 augimdsValidation = augmentedImageDatastore(inputSize(1:2),imdsValidation);
```
- Command Window:** Shows the execution of the script, with callout 1 pointing to the execution command.

```
>> % 4) Train Network with initial layers frozen to save time
% 4a) Freeze the first 10 layers to save processing time
layers = lgraph.Layers;
connections = lgraph.Connections;
layers(1:10) = freezeWeights(layers(1:10));
lgraph = createLgraphUsingConnections(layers,connections);
% Use an augmented image
imageAugmenter = imageDataAugmenter;
augimdsTrain = augmentedImageDatastore(inputSize(1:2),imdsTrain, ...
'DataAugmentation',imageAugmenter);
% Automatically resize training images without any additional preprocessing
augimdsValidation = augmentedImageDatastore(inputSize(1:2),imdsValidation);
```
- Property Inspector:** Shows properties for `imageAugmenter` (callout 2), `augimdsTrain` (callout 3), `imageAugmenter` (callout 4), and `augimdsValidation` (callout 5). Callout 6 points to the `OutputSize` property of `augimdsValidation`, which is set to `[224,224]`.

Copy and paste on command prompt the step 4a) where: the 10 first layers are frozen to save classification processing time in this net already pretrained, create a “imageAugmenter,” and automatically resize training images for: training “augimdsTrain” and “augiValidation” to “244 x 244 pixels”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

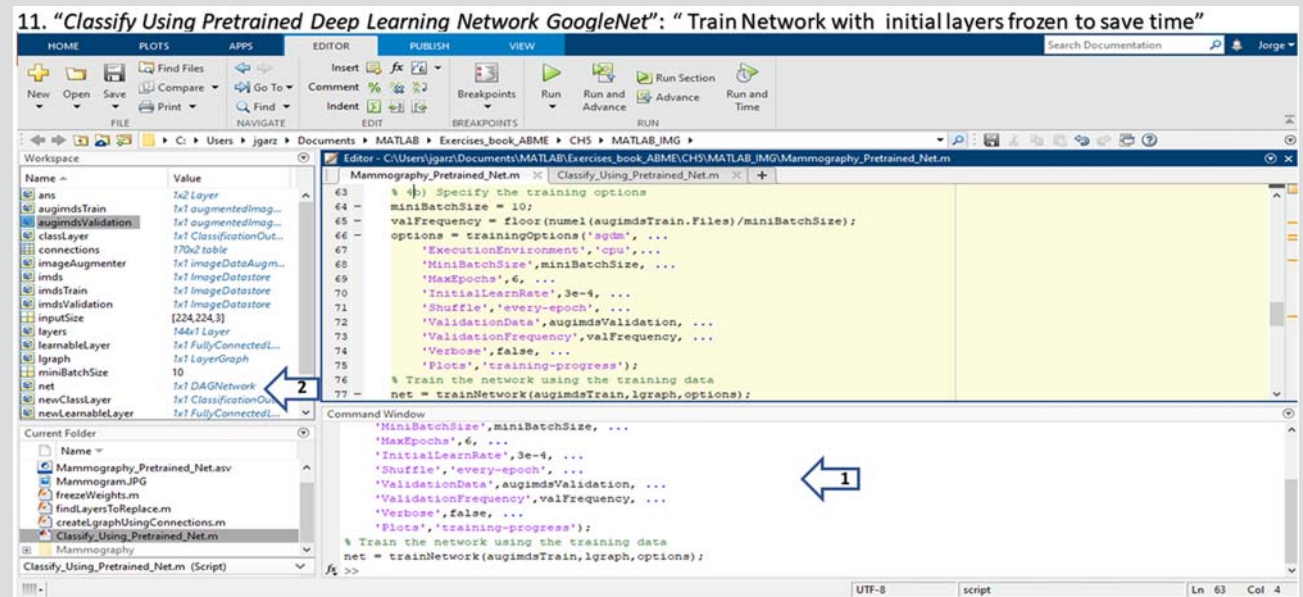
(Continued)

(Continued)

Slide Description

Screen figure

11 “Classify Using Pretrained Deep Learning Network GoogLeNet”: “Train Network with initial layers frozen to save time”
Copy and paste on command prompt the step 4a) where: the specification for the training are specified to a “single CPU” others options are “GPU that requires Parallel Computing Toolbox” and a “CUDA” (it is a programming model for GPU from ENVIDIA), with “6 Epochs,” plot “training-progress,” and others parameters

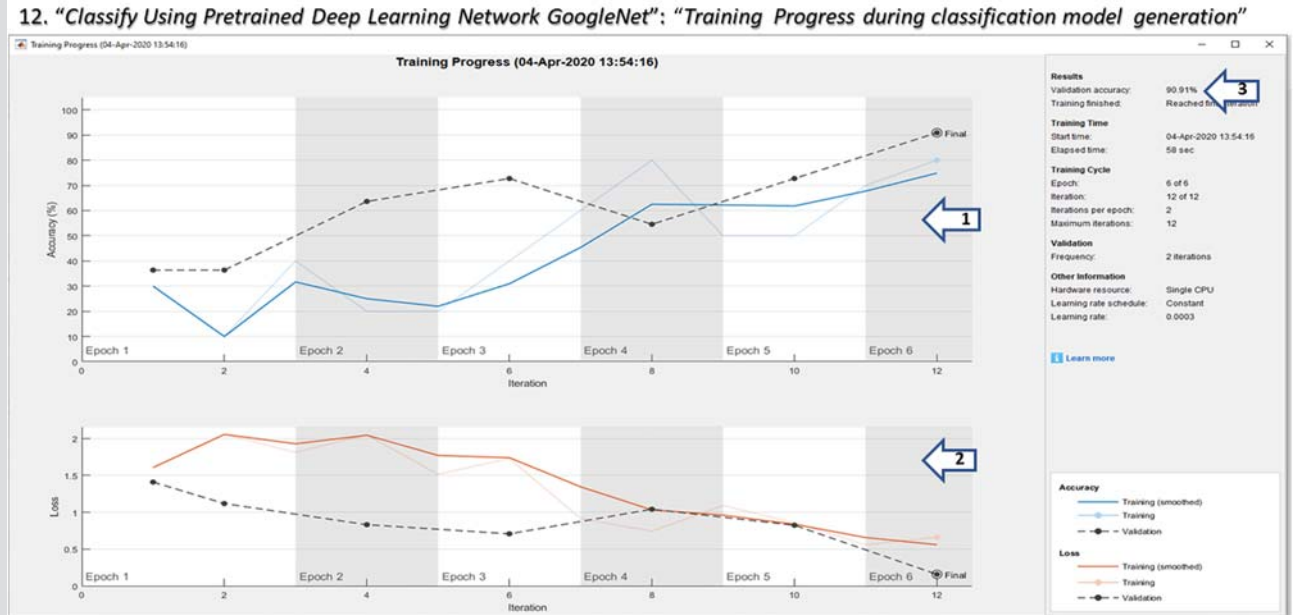


Copy and paste on command prompt the step 4a) where: the specification for the training are specified to a “single CPU” others options are “GPU that requires Parallel Computing Toolbox” and a “CUDA” (it is a programming model for GPU from ENVIDIA), with “6 Epochs,” plot “training-progress,” and others parameters.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

12

“Classify Using Pretrained Deep Learning Network GoogLeNet”:
“Training Progress during classification model generation”
The “training-progress” based on “6 Epoch—iterations” are shown in a plot of “Accuracy % versus Iterations” and “Loss versus Iterations” with a “Validation accuracy = 90.91%”



The “training-progress” based on “6 Epoch – iterations” are shown in a plot of “Accuracy % vs Iterations” and “Loss vs Iterations” with a “Validation accuracy = 90.91%”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

13 “Classify Using Pretrained Deep Learning Network GoogLeNet”: “Classify Validation images and Show random images with score”
Copy and paste on command prompt the step 5) where: the classify obtains the “Ypred” of 4 categories and store in “probs” their probabilities. Then a plot shows the 4 categories with their names and percentages probabilities

13. “Classify Using Pretrained Deep Learning Network GoogLeNet”: “Classify Validation images & Show random images with score”

```
5) Classify Validation images & Show random mamography is
[YPred,probs] = classify(net,augImdsValidation);
accuracy = mean(YPred == ImdsValidation.Labels);
idx = randperm(numel(ImdsValidation.Files),4);
figure
for i = 1:4
    subplot(2,2,i)
    I = readimage(ImdsValidation,idx(i));
    imshow(I)
    label = YPred(idx(i));
    title(string(label) + ", " + num2str(100*max(probs(idx(i),:))))
end
```

Command Window

```
>> 5) Classify Validation images & Show random images with score
[YPred,probs] = classify(net,augImdsValidation);
accuracy = mean(YPred == ImdsValidation.Labels);
idx = randperm(numel(ImdsValidation.Files),4);
figure
for i = 1:4
    subplot(2,2,i)
    I = readimage(ImdsValidation,idx(i));
    imshow(I)
    label = YPred(idx(i));
    title(string(label) + ", " + num2str(100*max(probs(idx(i),:))))
end
fx >>
```

Copy and paste on command prompt the step 5) where: the classify obtains the “Ypred” of 4 categories and store in “probs” their probabilities. Then a plot shows the 4 categories with their names and percentages probabilities

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

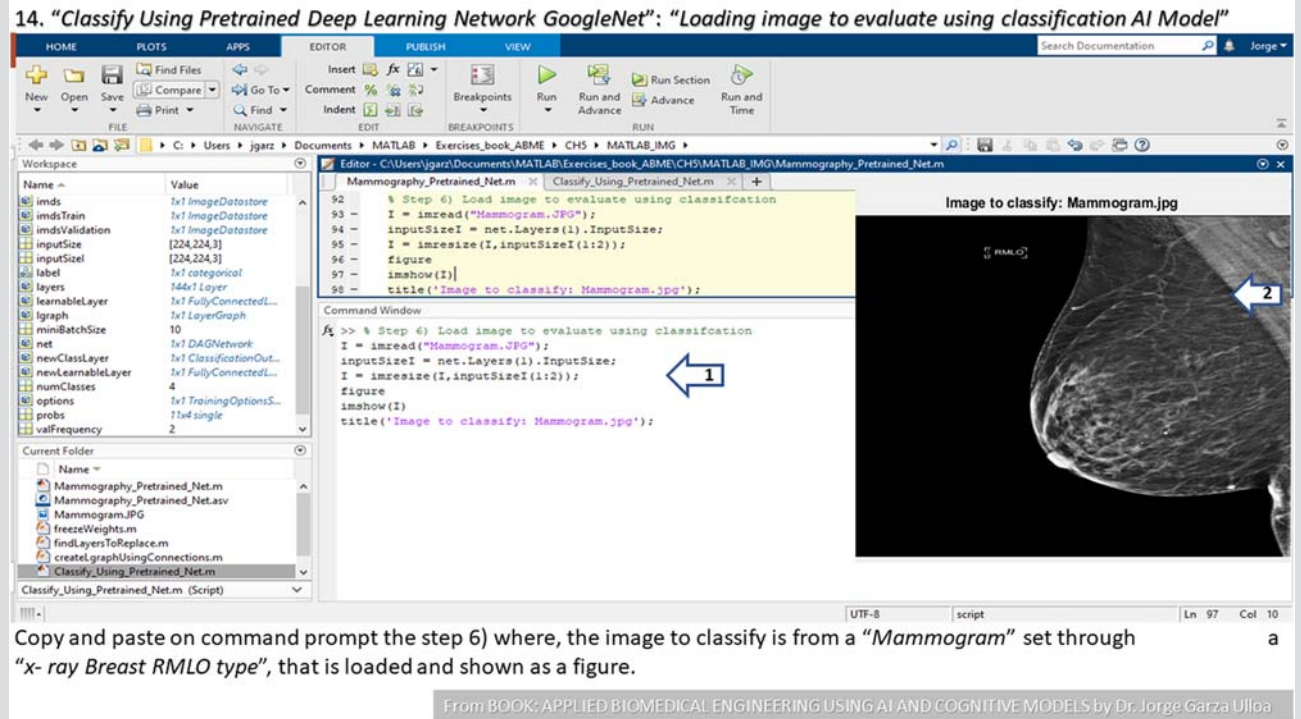
14

“Classify Using Pretrained Deep Learning Network GoogLeNet”:

“Loading image to evaluate using classification AI Model”

Copy and paste on command prompt the step 6) where, the image to classify is from a “Mammogram” set through a “X-ray Breast RMLO type,” that is loaded and shown as a figure.

14. “Classify Using Pretrained Deep Learning Network GoogleNet”: “Loading image to evaluate using classification AI Model”



The screenshot displays the MATLAB environment with the following components:

- Workspace:** Lists variables such as `imds`, `imdsTrain`, `imdsValidation`, `inputSize`, `inputSizeI`, `label`, `layers`, `learnableLayer`, `lgraph`, `minBatchSize`, `net`, `newClassLayer`, `newLearnableLayer`, `numClasses`, `options`, `probs`, and `valFrequency`.
- Current Folder:** Shows files including `Mammography_Pretrained_Net.m`, `Mammography_Pretrained_Net.asv`, `Mammogram.JPG`, `freezeWeights.m`, `findLayersToReplace.m`, `createL_graphUsingConnections.m`, `Classify_Using_Pretrained_Net.m`, and `Classify_Using_Pretrained_Net.m (Script)`.
- Editor:** Contains the script `Classify_Using_Pretrained_Net.m` with the following code:

```
52 % Step 6) Load image to evaluate using classification
93 I = imread("Mammogram.JPG");
94 inputSizeI = net.Layers(1).InputSize;
95 I = imresize(I, inputSizeI(1:2));
96 figure
97 imshow(I)
98 title('Image to classify: Mammogram.jpg');
```
- Command Window:** Shows the execution of the script:

```
% Step 6) Load image to evaluate using classification
I = imread("Mammogram.JPG");
inputSizeI = net.Layers(1).InputSize;
I = imresize(I, inputSizeI(1:2));
figure
imshow(I)
title('Image to classify: Mammogram.jpg');
```
- Figure Window:** Displays the image titled "Image to classify: Mammogram.jpg", showing a mammogram. A blue arrow labeled "2" points to the image.

Copy and paste on command prompt the step 6) where, the image to classify is from a “Mammogram” set through a “x- ray Breast RMLO type”, that is loaded and shown as a figure.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 15 Description
"Classify Using Pretrained Deep Learning Network GoogLeNet":
"Classify the new image and calculate its class probabilities"
Copy and paste on command prompt the step 7) where, the image is classified as "Breast RMLO with 98.2% of probability," and the top predictions are shown in a plot. Finally save workspace variable 'net' using the command: save('breast_net','net'), these variables could be useful later. Finally, close all

Screen figure

15 "Classify Using Pretrained Deep Learning Network GoogLeNet": "Classify the new image and calculate its class probabilities"

The screenshot displays the MATLAB environment with the following components:

- Editor:** Contains MATLAB code for classification. Line 100: `[label,scores] = classify(net,I);` Line 101: `figure;imshow(I);` Line 102: `title('GoogLeNet Mammogram = '+string(label) + ', ' + ...` Line 103: `num2str(100*scores(net.Layers(end).Classes == label),3) + '%')` Line 104: `% Show the four MAMMOGRAPHY predictions` Line 105: `num2str(100*scores(net.Layers(end).Classes == label),3) + '%')` Line 106: `[-,idx] = sort(scores,'descend');` Line 107: `idx = idx(4:-1:1);` Line 108: `classNamesTop = net.Layers(end).ClassNames(idx);` Line 109: `scoresTop = scores(idx);` Line 110: `figure` Line 111: `barh(scoresTop)` Line 112: `xlim([0 1])` Line 113: `title('Top 4 Predictions')` Line 114: `xlabel('Probability')` Line 115: `yticklabels(classNamesTop)` Line 116: `save('breast_net','net')`
- Command Window:** Shows the execution of the classification command: `classNamesTop = net.Layers(end).ClassNames(idx);` `scoresTop = scores(idx);` `figure` `barh(scoresTop)` `xlim([0 1])` `title('Top 4 Predictions')` `xlabel('Probability')` `yticklabels(classNamesTop)` `save('breast_net','net')`
- Workspace:** Lists variables such as `imageAugmenter`, `imds`, `imdsTrain`, `imdsValidation`, `inputSize`, `inputSize`, `label`, `layers`, `learnableLayer`, `lgraph`, `miniBatchSize`, `net`, `newClassLayer`, `newLearnableLayer`, `numClasses`, `options`, and `probs`.
- Figure:** A bar chart titled "Top 4 Predictions" showing the probability of four classes: Breast RMLO (98.2%), Breast LMLO, Breast LCC, and Breast RCC. The x-axis is labeled "Probability" and ranges from 0 to 1. The y-axis lists the classes. A blue bar for Breast RMLO is highlighted with a callout '3'. A callout '2' points to the mammogram image above the chart.

Copy and paste on command prompt the step 7) where, the image is classified as "Breast RMLO with 98.2% of probability", and the top predictions are shown in a plot. Finally save workspace variable 'net' using the command: save('breast_net','net'), these variables could be useful later. Finally, close all.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

The original “GoogLeNet” is a “convolutional neural network that is 22 layers deep, 144 layers total.” The network trained on “ImageNet classifies images into 1000 object categories, such as keyboard, mouse, pencil, and many animals.” The pretrained network have an image input size of “224-by-224.” An AI model using a modified “Deep Convolutional Neural Network (DCN)” based on a “Pretrained DCN as GoogLeNet to Classify Mammograms standard views types” was used with excellent classification for “mammography images.”

Recommendation

Using “Modified Pretrained Network” can be applied effortlessly to obtain “AI models” for images of many “Medical Image scanners” to analyze, classify and predict the scan images from the biology body [78] as: “X-ray,” “Computed tomography (CT)” that uses a computer-processed combination of many X-ray images taken from different angles, to produce cross-sectional images as virtual slices from the body, “Magnetic Resonance Imaging (MRI)” that uses strong magnetic fields, radio waves, and field gradients to form body images, “Functional Magnetic Resonance Imaging (fMRI)” that use “MRI” technology that measure brain activity by detecting changes associated with blood flow, “Ultrasound imaging” that is a diagnostic imaging technique based on the application of ultrasound as the “Doppler ultrasound Analyzer,” and many more.

5.5.2 Research 5.6 modify a “Pretrained Deep Convolutional Neural Network” to obtain an AI model to “classify Mammograms view type and suggest breast abnormalities as possible breast tumor”

5.5.2.1 Case for research

Obtain an AI using a “Deep Convolutional Neural Network (DCN) model through the MATLAB Deep Network Designer,” based on a “Modified Pretrained DCN as GoogLeNet to Classify Mammograms standard views types and suggest breast abnormalities as possible breast tumor or breast normal.”

5.5.2.2 General objective

Apply “MATLAB Deep Learning Toolbox” using “Deep Network Designer” to define, build, train, and deploy a “Deep Convolutional Neural Network (DCN) model” based on a “Modified Pretrained DCN as GoogLeNet to Classify Mammograms standard views types and suggest breast abnormalities as possible breast tumor or breast normal.”

5.5.2.3 Background “Mammograms to detect breast abnormalities”

The image of the breast is taken by a “low-dose X-rays” from bioinstruments known as a “mammogram,” and the

images are taken in a black background then the breast is show up in grays and whites. Then, denser tissues, connective tissues and glands show up in white. Any area that does not look like a normal tissue are areas of white as high-density tissue, and It is important to take note of its size, shape, and edges. A “lump” or “tumor” shows up as a white area on a “mammogram.” Breast abnormalities are usually “mass” detected in a breast could be: “tumor,” “cyst,” “calcifications,” “fibroadenomas,” and “scar tissues,” where:

- “Tumor” is a mass that can be “cancerous” or “benign.” When the “tumor” is “benign” is considered unlikely to grow and change shape, and “cancerous” when is likely to grow and change shape.
- “Cyst” are small fluid-filled sacs. Most are simple “cysts,” which have a thin wall and are not cancerous.
- “Calcifications” are deposits of “calcium,” if there are small deposits are known as “microcalcifications,” if there are larger deposits are identified as “macrocalcifications.” It is important based on their appearances be tested for possible sign of “cancer.”
- “Fibroadenomas” are “benign tumors” in the breast. They are round and they can occur at any age.
- “Scar tissues” often appears within on “mammogram.”

Others important breast abnormalities to detect on “mammograms” are: “Small white specks,” and “Breast density,” where:

- “Small white specks” are usually harmless, but it is important to check their shape and pattern, because they can sometimes be a sign of “cancer.”
- “Breast density” could be a sign of “slightly higher risk of breast cancer.”

There is a standard system to evaluate a “mammogram,” and it known as “Breast Imaging-Reporting and Data System (BIRADS)” and it has seven categories with numbers from 0 to 6:

1. “BIRADS category 0” is when the result is unclear, and more test or comparison with previous “mammograms” are needed.
2. “BIRADS category 1” is when no abnormalities are found.
3. “BIRADS category 2” is when images show abnormalities as “benign calcifications” but no signs of “cancer.”
4. “BIRADS category 3” is when images show abnormalities as “benign calcifications” but need “following up to discard signs of cancer.”
5. “BIRADS category 4” is when images show abnormalities that could be “cancerous,” possibly requiring a “biopsy.”
6. “BIRADS category 5” is when images show abnormalities that are highly likely to be “cancerous,” requiring a “biopsy.”
7. “BIRADS category 6” is when images show that “cancer” is present, requiring more “mammograms” to check progress.

A radiologist often compares an actual “mammogram” against previous images to detect change on “spots” and request a “biopsy,” that is an examination of “breast tissue from that spots” to decide if that unusual area could be a sign of “cancer.”

5.5.2.4 Specific objectives

- Load a “MATLAB net variable” from the “Pretrained DCN GoogLeNet” defined and save it from the workspace in the last research.
- Use the “MATLAB Deep Learning Toolbox” using “Deep Network Designer” to load the “net variable” based on the “Pretrained Network GoogLeNet.”
- Obtain from “Deep Network Designer” a “Net Diagram” and replace graphically two of the last layers for: “Fully Connected Convolution” and a new “Class Output.”
- Apply from “Deep Network Designer” the “Network Analyzer” to verify the two layers replaced are correct.
- Import from “Deep Network Designer” the data companion folder with the images dataset “Mammography_NT.”
- Plot the “Training data categories” and the “Validation data categories.”
- Specify the “Training Options” using the “Solver Adaptive Moment Estimation (ADAM)” as optimizer for the “DCN,” and train it and retrain it until obtain an “accuracy $\geq 66\%$ ” in the “AI model.”
- Export the trained network model as “MATLAB net” and “net structure information” as variables to the workspace.
- Load and test different “mammography” to classified them based on their “standard views types and suggest breast abnormalities as possible breast tumor or breast normal” and obtain the “five top predictions with their class probabilities.”

5.5.2.5 Dataset

The “low-dose X-ray” image dataset from screening “Mammography_NT” is organized in eight folders, four folders from patients with “normal breast in one folder for each standard view type,” and another four folders from patients with “breast abnormalities in one folder for each standard view type.” The folder name defines the mammography category with the following standard views name: “Breast Normal LCC,” “Breast Normal LMLO,” “Breast Normal RCC,” “Breast Normal RMLO,” “Breast Tumor LCC,” “Breast Tumor LMLO,” “Breast Tumor RCC,” and “Breast Tumor RMLO.” The folders organization and their respective images are shown in Fig. 5.22. Where each folder contains images from different patients with a “image size known as pixel resolution of 2048×2432 .”

5.5.2.6 Procedure

The steps to obtain an “AI Deep Convolutional Neural Network (DCN) model” through the “MATLAB Deep Network Designer,” based on a “Modified Pretrained DCN as GoogLeNet to Classify Mammograms standard views types and suggest breast abnormalities as possible breast tumor or breast normal” are summarized in Table of slides 5.6, and each step of the example are visually explained using screen sequences with instructions in easy to follow screen figures.

Note: In this research “any breast abnormalities are going to be classified as suggested possible breast tumor and calculate its probability, the final decision must be taken for a radiologist based on mammograms through time and biopsy results.”

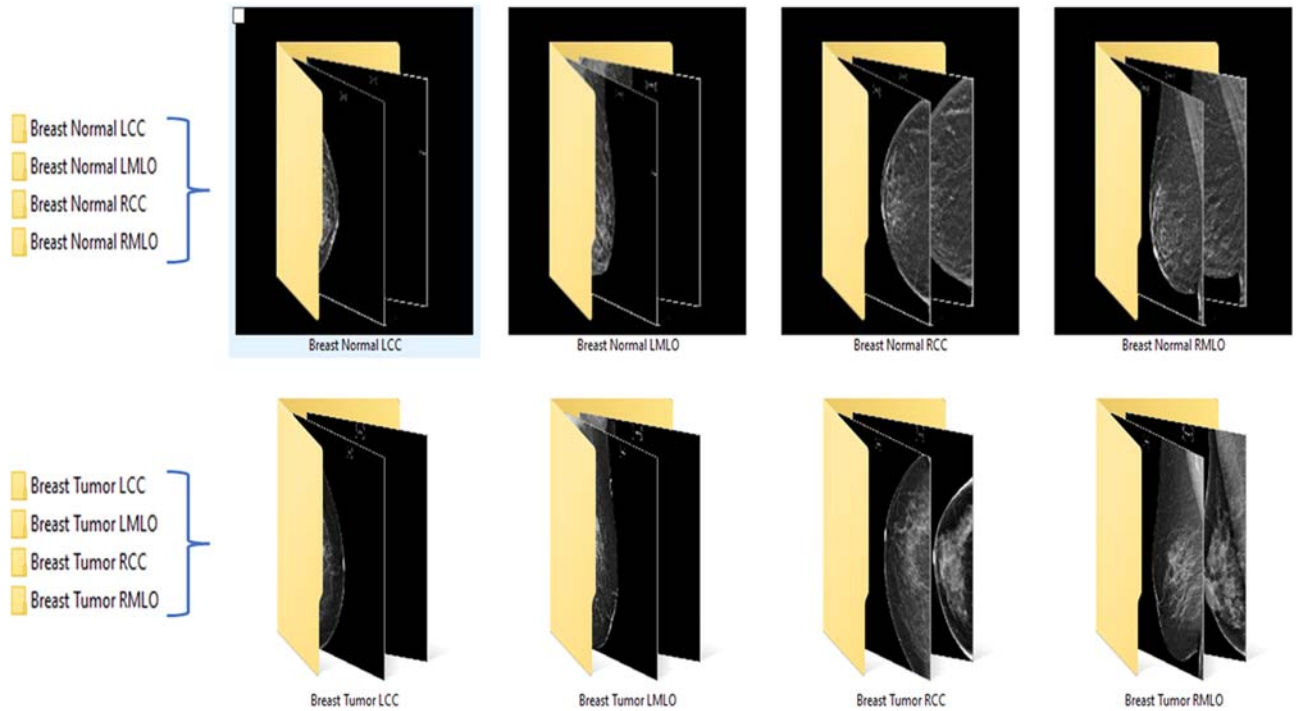


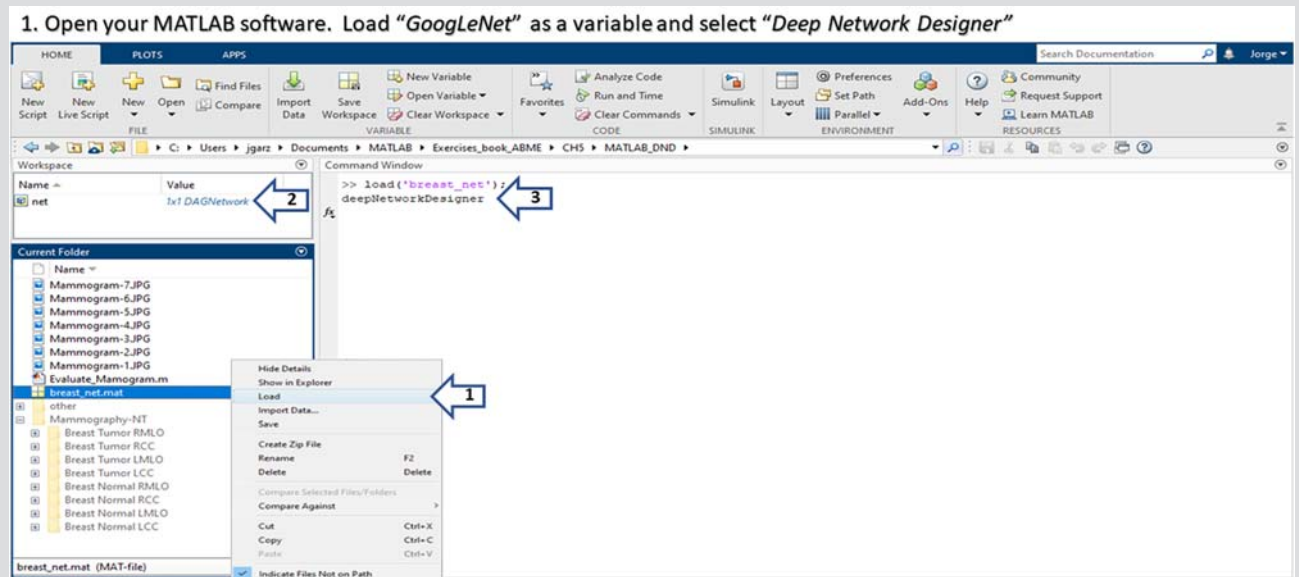
FIGURE 5.22 Image Dataset “*Mammography-NT*” is organized in eight folders according with the Mammograms standard views type. Note: This images dataset is available in the companion directory of the book, in the following directory: “*.. \Exercises_book_ABME\CH5\MATLAB_DND \Mammography-NT.*”

Table of slides 5.6 MATLAB Deep Learning Toolbox using “Deep Network Designer” to modify a “Pretrained Deep Convolutional Neural Network” to obtain an AI model to “classify Mammograms view type and suggest breast abnormalities as possible breast tumor or breast normal.”

Slide Description

Screen figure

1 Open your MATLAB software. Load “GoogLeNet” as a variable and select “Deep Network Designer”
 Go to the directory “... \Exercises_book_ABME\CH5 \MATLAB_DND,” and load the net workspace variable “breast_net.mat.” It contains the trained network of the last tutorial. Verify that the variable “net” is loaded in the “Workspace” and in the command prompt enter: “deepNetworkDesigner”



Go to the directory “... \Exercises_book_ABME\CH5\ MATLAB_DND”, and load the net workspace variable “breast_net.mat”. It contains the trained network of the last tutorial. Verify that the variable “net” is loaded in the “Workspace” and in the command prompt enter: “deepNetworkDesigner”

2

In the “MATLAB Deep Network Designer”: initial screen selecting “From Workspace, and GoogLeNet as Pretrained Network”

Verify that the pretrained Network: “GoogLeNet” is “loaded” (note: if there is triangle with the “! Sign,” it must be loaded first in the MATLAB Tab: Home > Add-Ons > Get Add-Ons), then click in the “General network: From Workspace”

2. In the “MATLAB™ Deep Network Designer”: initial screen selecting “From Workspace, and GoogLeNet as Pretrained Network”

Verify that the pretrained Network: “GoogLeNet” is “loaded” (note” if there is triangle with the “! Sign”, it must be loaded first in the MATLAB Tab: Home> Add-Ons> Get Add-Ons), then click in the “General network: From Workspace”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

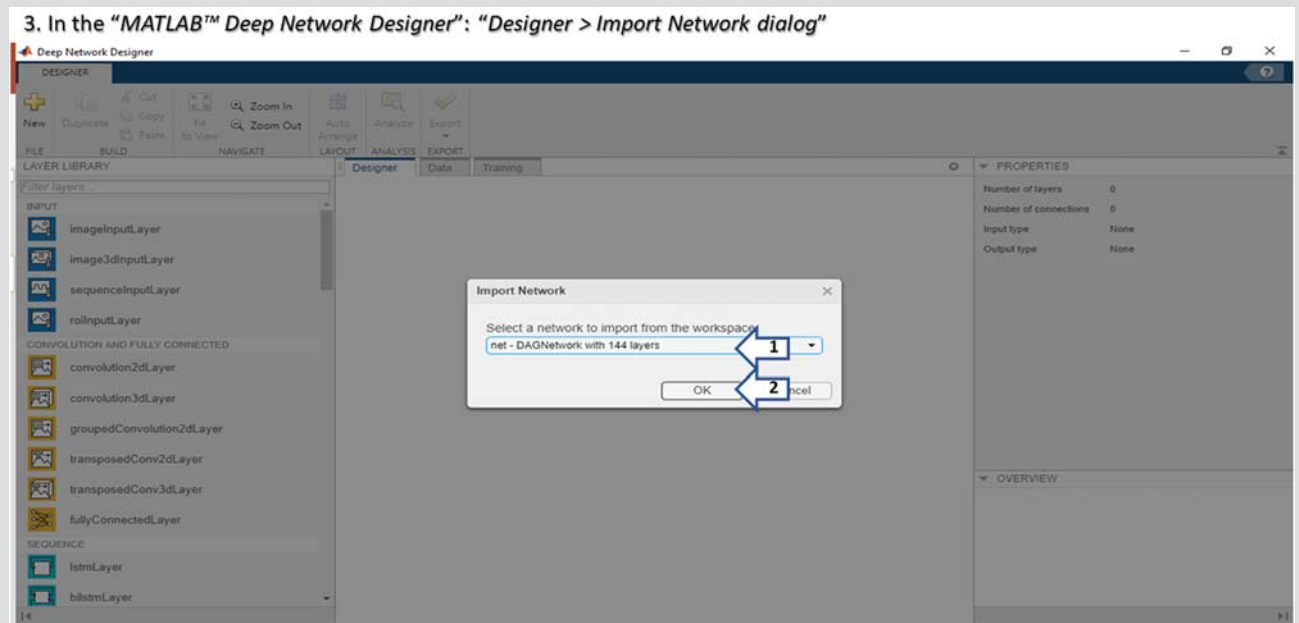
(Continued)

(Continued)

Slide **Description**

3 In the “MATLAB Deep Network Designer”: “Designer > Import Network dialog”
Select the variable previously loaded in the workspace as shown in slide 1: “net-DAGNetwork with 144 layers,” and press the button “Ok”

Screen figure

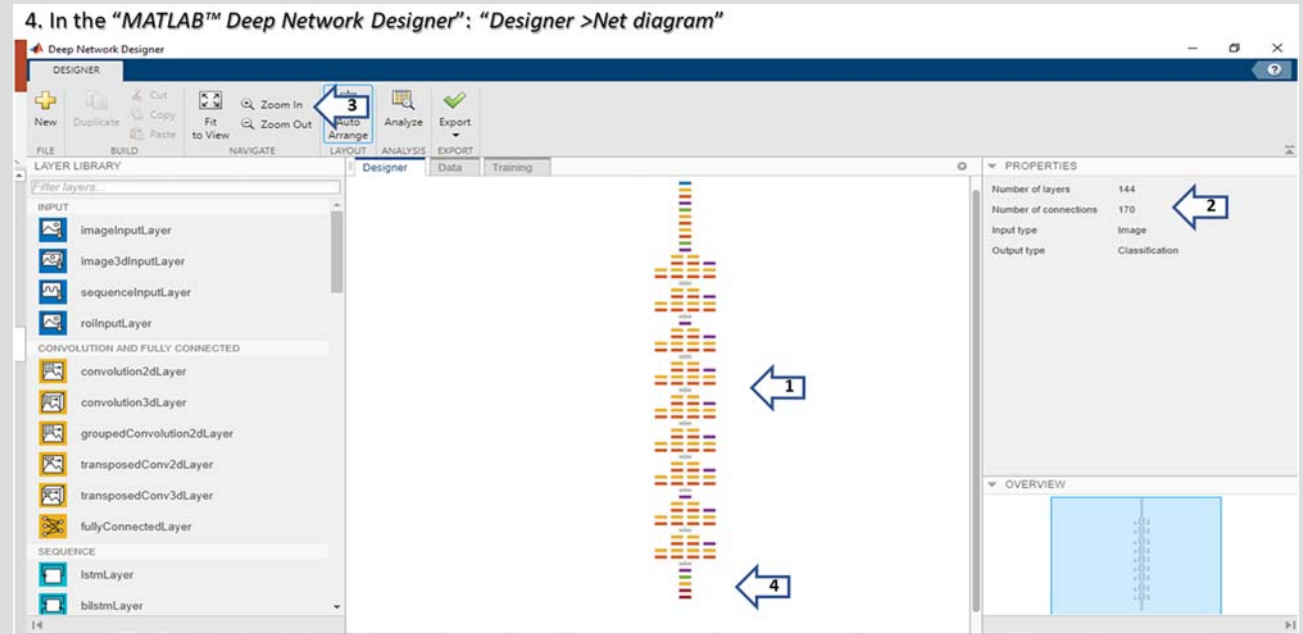


Select the variable previously loaded in the workspace as shown in slide 1: “net-DAGNetwork with 144 layers”, and press the button “Ok”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

4

In the “MATLAB Deep Network Designer”: “Designer > Net diagram”
Observe that this network was modified in the last research as a “Pretrained GoogLeNet” with “144 layers,” “170 connections,” with “images as input type,” and “Classification for output type.” Select the button “Zoom In: to amplify the last 6 layers”



Observe that this network was modified in the last research as a “Pretrained GoogLeNet” with “144 layers”, “170 connections”, with “images as input type”, and “Classification for output type”. Select the button “Zoom In: to amplify the last 6 layers”.

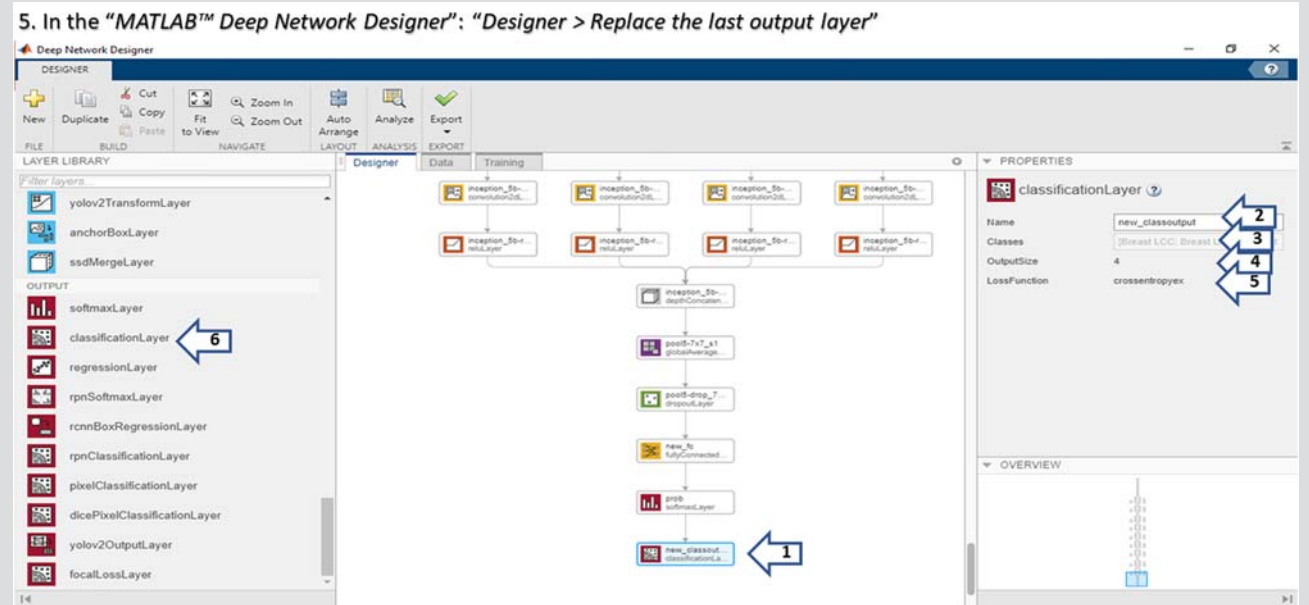
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 5
Description
In the "MATLAB Deep Network Designer": "Designer > Replace the last output layer"
Select last layer "classificationLayer" with name "new_classoutput," delete and replaced for a new classification output one with the name "classificationLayer," with "classes = [Breast LCC, Breast LMLO, Breast RCC, Breast RMLO]," "OutputSize = auto" and "loss function crossentropyex." and connect it again as shown in the next slide

Screen figure



Select last layer "classificationLayer" with name "new_classoutput", delete and replaced for a new classification output one with the name "classificationLayer", with "classes=[Breast LCC, Breast LMLO, Breast RCC, Breast RMLO]", "OutputSize=auto" and "loss function crossentropyex". and connect it again as shown in the next slide.

6

In the "MATLAB Deep Network Designer": Designer > New "classOutput layer" with "Classes = auto" and "OutputSize = auto"
The new output layer with "name = classoutput", "Classes = auto", "OutputSize = auto" and "LossFunction = crossentropyex".
Select the Layer "name = new_fxc", "InputSize = 1024", "OutputSize = 4", etc., delete it, replace with a "CONVOLUTION AND FULLY CONNECTED > fullyConnectLayer" and connect it again as shown in the next slide.

6. In the "MATLAB Deep Network Designer": Designer > New "classOutput layer" with "Classes=auto" and "OutputSize=auto"

The new output layer with "name=classoutput", "Classes=auto", "OutputSize=auto" and "LossFunction=crossentropyex". Select the Layer "name=new_fxc", "InputSize=1024", "OutputSize=4", etc., delete it, replace with a "CONVOLUTION AND FULLY CONNECTED> fullyConnectLayer" and connect it again as shown in the next slide.

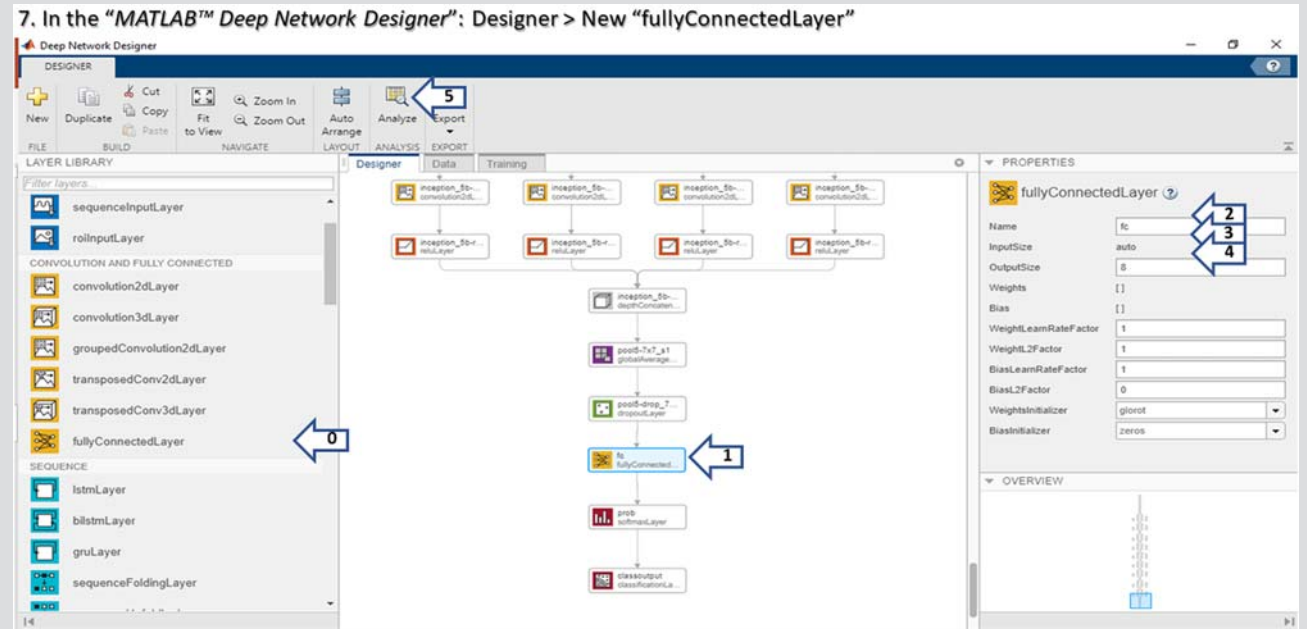
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 7 Description
In the "MATLAB Deep Network Designer": Designer > New "fullyConnectedLayer"
The new "CONVOLUTION AND FULLY CONNECTED > fullyConnectLayer" with "Name = fc," "InputSize = auto," "OutputSize = 8," Note: Leave the others parameter by their default. Click the button "Analyze" to verify that network is well configured

Screen figure



The new 'CONVOLUTION AND FULLY CONNECTED>fullyConnectLayer' with "Name=fc", "InputSize=auto", "OutputSize=8", Note: Leave the others parameter by their default. Click the button "Analyze" to verify that network is well configured.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

8

In the “MATLAB Deep Network Designer”: Network Analyzer”

If the network is well configured the network diagram is shown and the “ANALYSIS RESULT” for each of the “144 layers,” with “zero warnings” and “zero errors.” Got to the bottom of the network diagram and check “ANALYSIS RESULT” for the replaced layers.” Close “network Analyzer”

8. In the “MATLAB™ Deep Network Designer”: Network Analyzer”

The screenshot displays the MATLAB Deep Learning Network Analyzer interface. On the left, a network diagram shows a sequence of layers: data, conv1-relu, pool1-3x3, conv2-3x3, pool2-3x3, and a series of inception blocks. On the right, the 'ANALYSIS RESULT' table lists 144 layers with their respective types, activations, and learnables. Blue arrows point to specific layers in both the diagram and the table: arrow 1 points to the 'data' layer, arrow 2 to 'conv1-relu_s2', arrow 3 to the 'fc' layer, and arrow 4 to 'fc' in the table. Arrow 5 points to 'classoutput' in the diagram, and arrow 6 points to 'classoutput' in the table.

Name	Type	Activations	Learnables
1 data	Image Input	224x224x3	-
2 conv1-relu_s2	Convolution	112x112x64	Weights 7x7x3x64 Bias 1x1x64
3 conv1-relu_7x7	ReLU	112x112x64	-
4 pool1-3x3_s2	Max Pooling	56x56x64	-
5 pool1-norm1	Cross Channel Nor...	56x56x64	-
6 conv2-3x3_reduce	Convolution	56x56x64	Weights 1x1x64x64 Bias 1x1x64
7 conv2-relu_3x3_reduce	ReLU	56x56x64	-
8 conv2-3x3	Convolution	56x56x192	Weights 3x3x64x192 Bias 1x1x192
9 conv2-relu_3x3	ReLU	56x56x192	-
10 conv2-norm2	Cross Channel Nor...	56x56x192	-
11 pool2-3x3_s2	Max Pooling	28x28x192	-
120 inception_5b-relu_3x3	ReLU	7x7x384	-
127 inception_5b-relu_pool_proj	ReLU	7x7x128	-
130 inception_5b-relu_1x1	ReLU	7x7x384	-
132 inception_5b-output	Depth concatenation	7x7x1024	-
140 pool5-7x7_s1	Global Average Po...	1x1x1024	-
141 pool5-drop_7x7_s1	Dropout	1x1x1024	-
142 fc	Fully Connected	1x1x8	Weights 8x1024 Bias 8x1
143 prob	Softmax	1x1x8	-
144 classoutput	Classification Output	-	-

If the network is well configured the network diagram is shown and the “ANALYSIS RESULT” for each of the “144 layers”, with “zero warnings” and “zero errors”. Got to the bottom of the network diagram and check “ANALYSIS RESULT” for the replaced layers”. Close “network Analyzer”

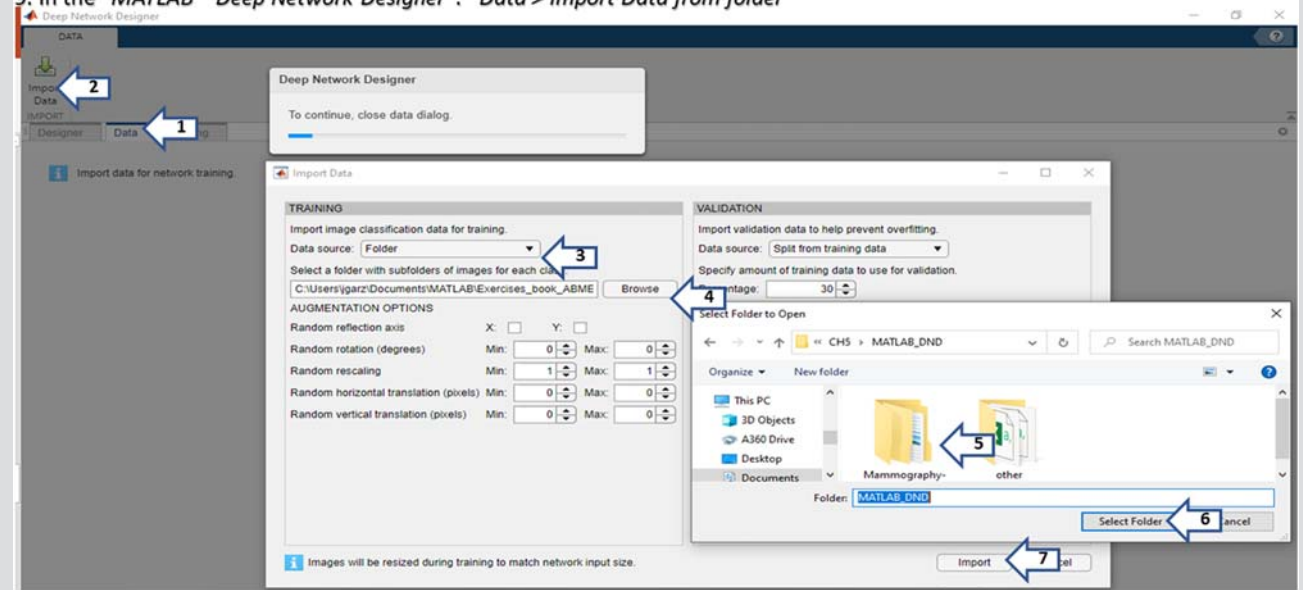
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 9 Description
In the "MATLAB Deep Network Designer": "Data > Import Data from folder"
In the "MATLAB Deep Network Designer" select the "Data" tab, then click on "Import Data" icon, select on "TRAINING: Data source = Folder," click on the "Browse" button and select the folder "Mammography_NT" and finally the "Import" button. Note: Leave in Validation = 30%

Screen figure
9. In the "MATLAB™ Deep Network Designer": "Data > Import Data from folder"



In the "MATLAB™ Deep Network Designer" select the "Data" tab, then click on "Import Data" icon, select on "TRAINING: Data source=Folder", click on the "Browse" button and select the folder "Mammography_NT" and finally the "Import" button. Note: Leave in Validation=30%

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

10

In the “MATLAB Deep Network Designer”: “Data > Import Data— Training plot”

A plot is generated for the selected “training images classes versus number of them in each class.” “They are a total of 51 images (observations), in eight classes from 5 to 8 observations in each category”



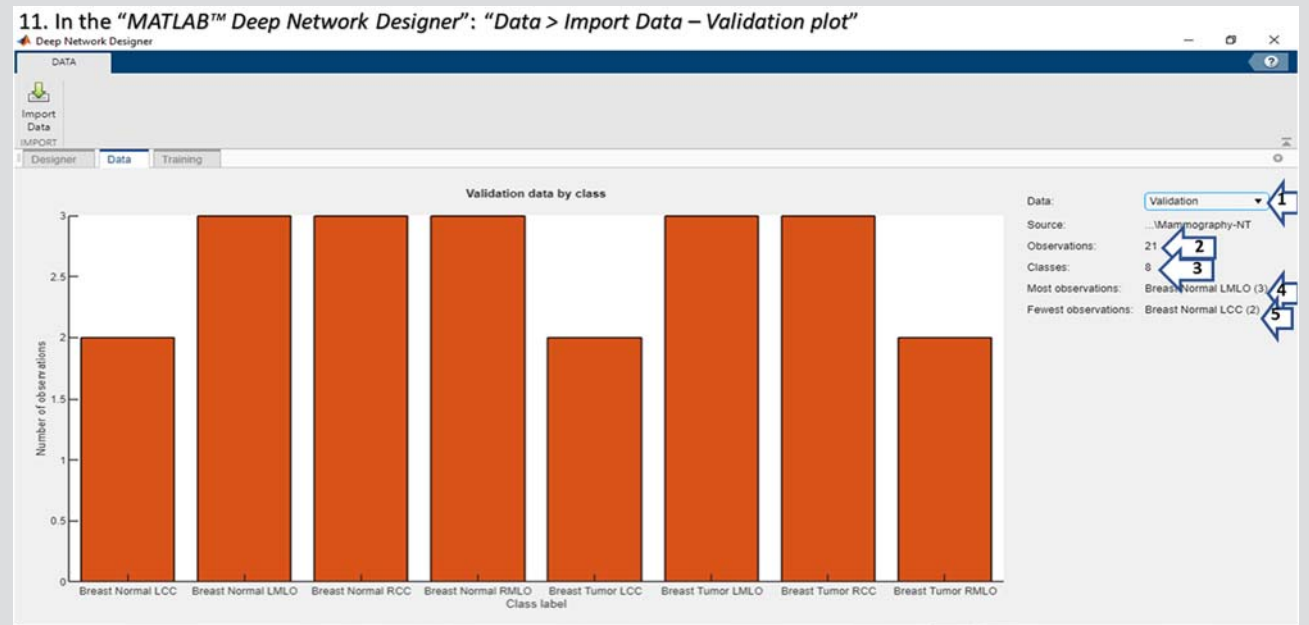
(Continued)

(Continued)

Slide **Description**

11 In the “MATLAB Deep Network Designer”: “Data > Import Data – Validation plot”
A plot is generated for the selected “validation images classes versus number of them in each class.” “They are a total of 21 images for validation (30% of observations), in eight classes from 2 to 3 observations in each category”

Screen figure



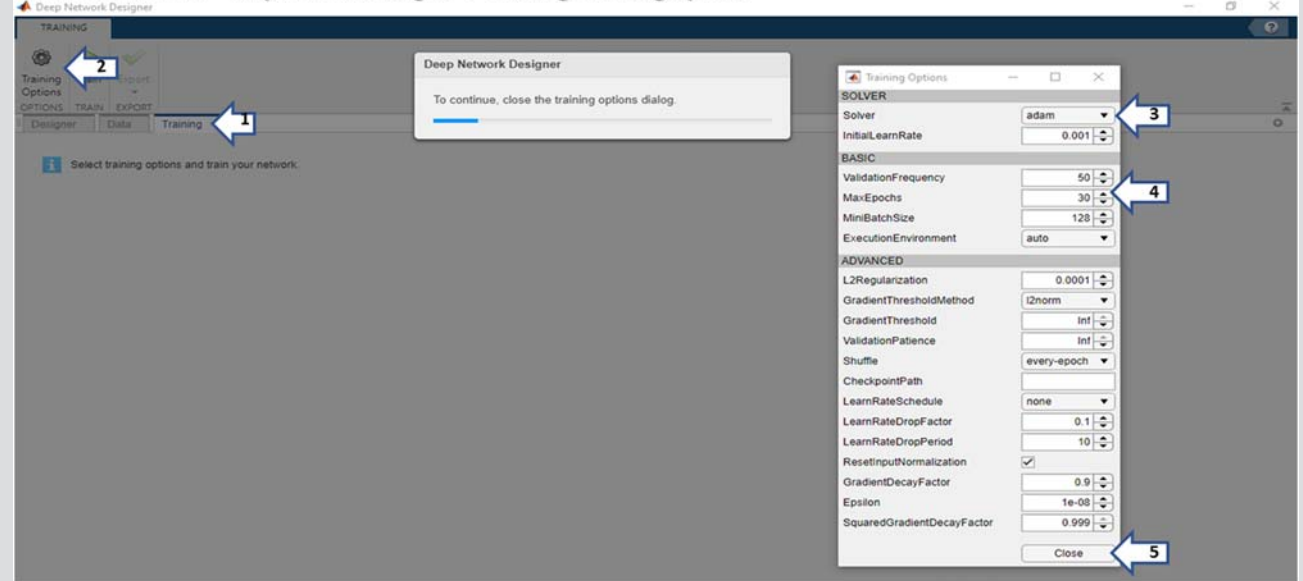
A plot is generated for the selected “validation images classes vs number of them in each class”. “They are a total of 21 images for validation (30% of observations), in eight classes from 2 to 3 observations in each category”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

12

In the "MATLAB Deep Network Designer": "Training > Training Options"
Click on "Training" tab, then on "Training Options." In the dialog screen select: "Solver = adam," "MaxEpochs = 30," and click on "Close" button"

12. In the "MATLAB™ Deep Network Designer": "Training > Training Options"



Click on "Training" tab, then on "Training Options". In the dialog screen select: "Solver = adam," "MaxEpochs = 30", and click on "Close" button".

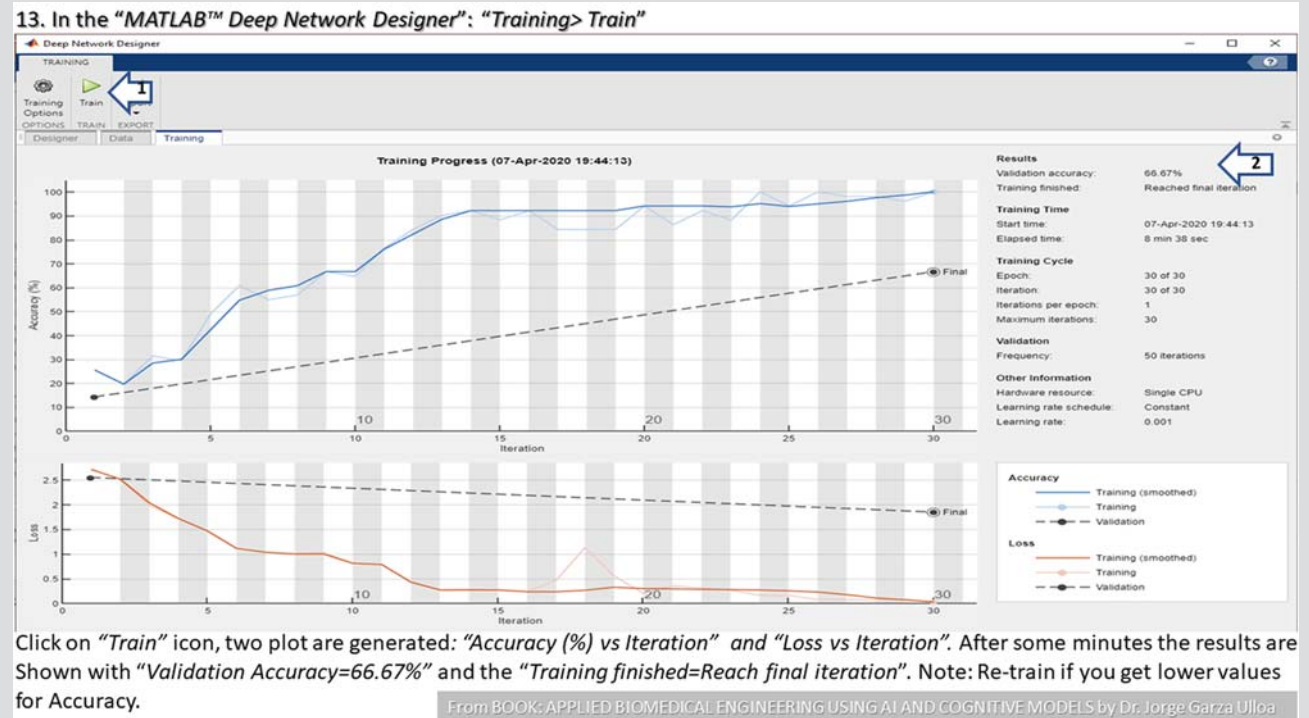
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 13 Description In the "MATLAB Deep Network Designer": "Training > Train" Click on "Train" icon, two plots are generated: "Accuracy (%) versus Iteration" and "Loss versus Iteration." After some minutes the results are shown with "Validation Accuracy = 66.67%" and the "Training finished = Reach final iteration." Note: Retrain if you get lower values for Accuracy

Screen figure



14

In the "MATLAB Deep Network Designer": "Training > Export"
Click on "Export" button, then on
"Export Trained Network and Results"

14. In the "MATLAB™ Deep Network Designer": "Training > Export"



Click on "Export" button, then on "Export Trained Network and Results"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 15 Description In MATLAB workspace—“Script: Evaluate_Mammogram.m-Step 1) Input dialog to load image to classify” Verify that the Workspace has three variables: “Net,” “trained_Network1,” and “trainInfoStruct_1.” Load the script “Evaluate_Mammogram.m.” Copy and paste the step 1) Input dialog to load image to classify, and a figure with the image to be evaluated

Screen figure

The screenshot displays the MATLAB environment with the following components:

- Workspace:** Shows three variables: 'net' (1x1 DAGNetwork), 'trainedNetwork_1' (1x1 DAGNetwork), and 'trainInfoStruct_1' (1x1 struct). Arrows labeled 1 and 2 point to these variables.
- Current Folder:** Lists various image files (e.g., Mammogram-1.JPG) and folders (e.g., Mammography-NT). An arrow labeled 3 points to the 'Evaluate_Mammogram.m' script.
- Editor:** Shows the script 'Evaluate_Mammogram.m' with the following code:

```
1 %%Classify Image Using Pretrained Network
2 % Chap. 5 APPLIED BIOMEDICAL ENGINEERING
3 % Elsevier-Academic Press by Jorge Garza Ulloa
4 %% Step 1) Input dialog to load image to
5 - [fileName,filePath] = uigetfile('*.jpg',
6 - 'Select data file', '.'); % Input dig
7 - if filePath==0, error('None selected'); end % verify image is selected
8 - I = imread( fullfile(filePath,fileName) ); % Read image
9 - inputSize = trainedNetwork_1.Layers(1).InputSize;
10 - I = imresize(I,inputSize(1:2));
11 - figure
12 - imshow(I)
13 - title('Image to classify: Mammogram');
```

Arrows labeled 4 and 5 point to the 'uigetfile' function call and the 'Open' button in the dialog, respectively.
- Select data file dialog:** Shows the file selection process. An arrow labeled 4 points to the selected image file 'Mammogram-1.JPG', and an arrow labeled 5 points to the 'Open' button.
- Command Window:** Shows the execution of the script, resulting in the following output:

```
>> %% Step 1) Input dialog to load image to classify
[fileName,filePath] = uigetfile('*.jpg', 'Select data file', '.'); % Input dig
if filePath==0, error('None selected'); end % verify image is selected
I = imread( fullfile(filePath,fileName) ); % Read image
inputSize = trainedNetwork_1.Layers(1).InputSize;
I = imresize(I,inputSize(1:2));
figure
imshow(I)
title('Image to classify: Mammogram');
```

An arrow labeled 6 points to the 'title' line in the Command Window output.
- Figure:** Displays the loaded mammogram image with the title 'Image to classify: Mammogram'. An arrow labeled 6 points to the image.

Verify that the Workspace has 3 variables: “Net”, “trained_Network1”, and “trainInfoStruct_1”. Load the script “Evaluate_Mammogram.m”. Copy and paste the step 1) Input dialog to load image to classify, and a figure with the image to be evaluated.

16

In MATLAB workspace—“Script: Evaluate_Mammogram.m-Step 2) Classify the image and calculate the class probabilities”
Copy and paste the step “2) Classify the image and calculate the class probabilities,” and a figure with the image classified and evaluated is shown with as a suggested “Right Breast Normal RMLO with 100% of probability”

16. In MATLAB workspace – “Script: Evaluate_Mammogram.m - Step 2) Classify the image and calculate the class probabilities”

The screenshot shows the MATLAB environment with the following components:

- Workspace:** Lists variables such as `classNames` (categorical), `fileName` ('Mammogram-1.JPG'), `filePath` ('C:\Users\jgarz\Docu...'), `inputSize` ([224,224,3]), `label` (1x1 categorical), and `net` (1x1 DAGNetwork).
- Current Folder:** Shows a directory structure including 'Mammogram-7.JPG' through 'Mammogram-1.JPG', 'Evaluate_Mammogram.m', 'breast_net.mat', and a subfolder 'other' containing 'Mammography-NT' and 'Breast Tumor RMLO' through 'Breast Normal RMLO'.
- Editor:** Contains the script `Evaluate_Mammogram.m` with the following code:

```
14 % Step 2) Classify the image and calculate the class probabilities
15 [label,scores] = classify(trainedNetwork_1,I);
16 figure
17 imshow(I)
18 classNames =trainedNetwork_1.Layers(end).Classes;
19 title("Classify MAMMOGRAPHY as "+string(label) + ", " + ...
20 num2str(100*scores(classNames == label),3) + "%");
21 disp(classNames(label))
```
- Command Window:** Shows the execution of the script:

```
>> % Step 2) Classify the image and calculate the class probabilities
[label,scores] = classify(trainedNetwork_1,I);
figure
imshow(I)
classNames =trainedNetwork_1.Layers(end).Classes;
title("Classify MAMMOGRAPHY as "+string(label) + ", " + ...
num2str(100*scores(classNames == label),3) + "%");
disp(classNames(label))
Breast Normal RMLO
```
- Figure Window:** Displays the classified image with the title "Classify MAMMOGRAPHY as Breast Normal RMLO, 100%".

Copy and paste the step “2) Classify the image and calculate the class probabilities”, and a figure with the image classified and evaluated is shown with as a suggested “Right Breast Normal RMLO with 100% of probability”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 17 Description
MATLAB workspace "Script: Evaluate_Mammogram.m-Step 3) Display Top Predictions classifications using GoogleNet modified"
Copy and paste the step "3) Display Top Predictions classifications using GoogleNet modified," and a figure with the plot for 8 classes available for "Mammograms" is shown suggesting a "Right Breast Normal RMLO with 100% of probability" and "0% of probability for the others classes"

Screen figure

17. MATLAB workspace "Script: Evaluate_Mammogram.m-Step 3) Display Top Predictions classifications using GoogleNet modified"

The screenshot shows the MATLAB environment with a script editor and a command window. The script editor displays the following code:

```
23 % Step 3)Display Top Predictions classification
24 [-,idx] = sort(scores,'descend');
25 idx = idx(8:-1:1);
26 classNamesTop = classNames(idx);
27 scoresTop = scores(idx);
28 figure
29 barh(scoresTop)
30 xlim([0 1])
31 title('Top 5 Predictions')
32 xlabel('Probability')
33 yticklabels(classNamesTop)
```

The command window shows the execution of the script:

```
>> % Step 3)Display Top Predictions classifications
[-,idx] = sort(scores,'descend');
idx = idx(8:-1:1);
classNamesTop = classNames(idx);
scoresTop = scores(idx);
figure
barh(scoresTop)
xlim([0 1])
title('Top 5 Predictions')
xlabel('Probability')
yticklabels(classNamesTop)
```

The figure titled "Top 5 Predictions" is a horizontal bar chart showing the probability distribution for the top 5 predicted classes. The x-axis is labeled "Probability" and ranges from 0 to 1. The y-axis lists the classes. The bars are as follows:

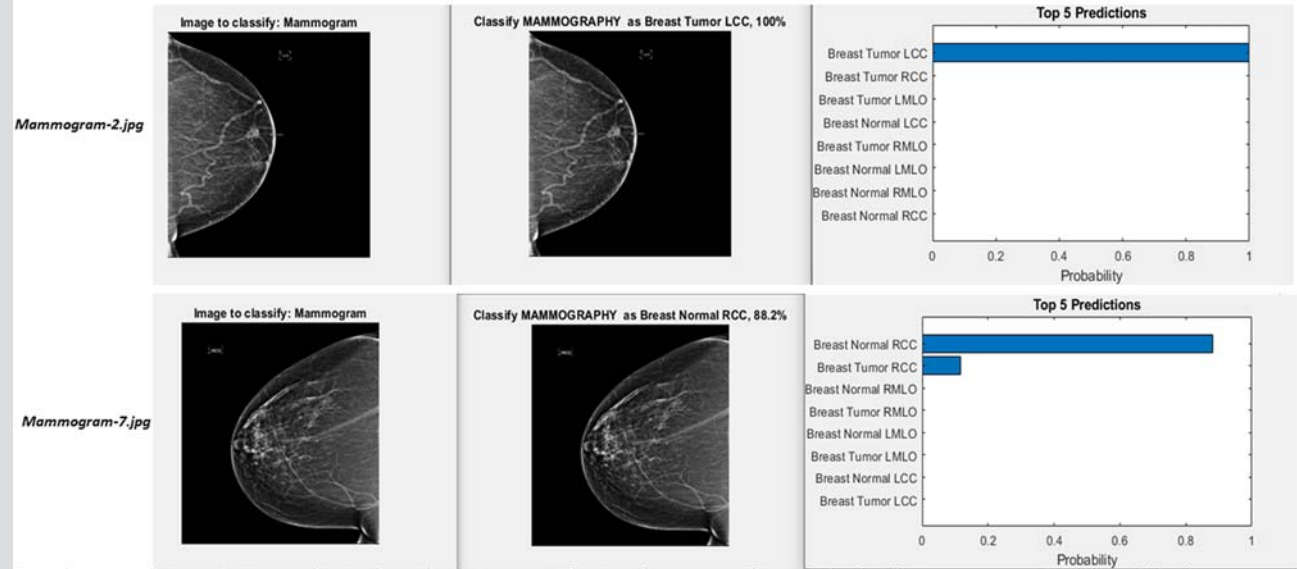
Class	Probability
Breast Normal RMLO	1.00
Breast Normal RCC	0.00
Breast Tumor RCC	0.00
Breast Normal LMLO	0.00
Breast Tumor RMLO	0.00
Breast Normal LCC	0.00
Breast Tumor LMLO	0.00
Breast Tumor LCC	0.00

Copy and paste the step "3) Display Top Predictions classifications using GoogleNet modified", and a figure with the plot for 8 classes available for "Mammograms" is shown suggesting a "Right Breast Normal RMLO with 100% of probability" and "0% of probability for the others classes".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

MATLAB workspace “Script: Evaluate_Mammogram. m” run for others breast mammograms Run the complete script to evaluate the other images. The top figures are the results for “Mammogram-2.jpg” and the lower figures are for “Mammogram-7.jpg.” Finally save all variables to a MAT-file for this in the “Home tab” in the variable section, click “Save Workspace” and close all open figures

18. MATLAB workspace “Script: Evaluate_Mammogram. m” run for others breast mammograms



Run the complete script to evaluate the others images. The top figures are the results for “Mammogram-2.jpg” and the lower figures are for “Mammogram-7.jpg” . Finally save all variables to a MAT-file for this in the “Home tab” in the variable section, click “Save Workspace” and close all open figures.

Conclusions

Applying “MATLAB Deep Learning Toolbox” using “Deep Network Designer” tool allows to define, build, train, and deploy an AI “Deep Convolutional Neural Network (DCN) model” based on a “Modified Pretrained DCN as GoogLeNet” to “Classify Mammograms standard views types and suggest breast abnormalities as possible breast tumor or breast normal.” The breast abnormalities are: “tumor,” “cyst,” “calcifications,” “fibroadenomas” “scar tissues” “Small white specks,” and “Breast density.” The DCN classification model results can help radiologist to simplify their works.

Recommendations

- Increase the number of “mammograms” used on training, validation, and test to increase accuracy.
- Apply this technique to detect other abnormalities on the human body as: “physical lesions,” “fractures,” “cancer spread,” etc.
- Create AI solutions to analyze “series of mammograms trough monthly intervals of times” of the same patient to detect is the “tumors are cancerous or not.”

5.5.3 Research 5.7 “custom Deep Convolutional Neural Network” to obtain an AI model to “classify Cervical X-rays view types”

5.5.3.1 Case for research

MATLAB Deep Learning Toolbox using “Deep Network Designer” to create a “custom Deep Convolutional Neural Network” to obtain an “AI model” to “classify cervical X-rays view types.”

5.5.3.2 General objective

Apply “MATLAB Deep Learning Toolbox” using “Deep Network Designer” to create a “custom Deep Convolutional Neural Network” to create, define, train, and deploy an “AI model” to “classify cervical X-rays view types.”

Note: This research is an introductory example with the main purpose of familiarize the reader on how to develop more endless different “Deep Neural Network” models for a diversity of Biomedical problems where an “AI Model” can help in many to analyze, classify, predict, detect abnormalities, and many other applications.

5.5.3.3 Background

“Cervical spine X-ray or C-Spine” are biomedical images usually purpose of detect “neck pain or trauma,” to find reasons for “upper limb weakness, numbness, or tingling”

[79]. The “standard CS spine X-rays” is a set of three views taken: “anteroposterior view,” “lateral view,” and “peg view.” Where:

- “Anteroposterior view” captures the “spine” from the front.
- “Lateral view” captures the image from the “spine” from one side.
- “Peg view” captures the upper part of the “cervical spine” and requires the patient to open the mouth wide.

“C-Spine X-Rays” are taken with the patient’s head in “full flexion.” The patient is asked to bend the head forward as far as possible, and to extend the neck backwards as far as possible.

5.5.3.4 Specific objectives

- Use “MATLAB Deep Network Designer” from blank network.
- Import from “Deep Network Designer” the folder with the images dataset “X-rays C-Spine.”
- Plot the “Training data categories” and the “Validation data categories” obtained from the dataset folders.
- Design step by step a “custom Deep Convolutional Neural Network (DCN)” using the “Deep Network Designer” components.
- Analyze the “custom Deep Convolutional Neural Network (DCN)” designed using the “Deep Network Designer” to find possible errors in the “Neural network.”
- Specify the “Training Options” using the “Solver Adaptive Moment Estimation (ADAM)” as optimizer for the “DCN,” and train it and retrain it if necessary, until obtain an “accuracy $\geq 66\%$ ” in the “AI model.”
- Export the trained network model as MATLAB “net” and “net structure information” as variables to the MATLAB workspace.
- Load and test different “X-rays C-Spine” images to classified them based on their standard views types and obtain the “their class probabilities.”

5.5.3.5 Dataset

The “X-rays C-Spine” images dataset is organized in three folders; in each folder there is 4 images of the same view type as explained in the last section, these are: “C-Spine Anteroposterior View,” “C-Spine Lateral View,” and “C-Spine Peg View.” These folders and images with picture size of resolution of “227 × 227” pixels, they are shown in Fig. 5.23:

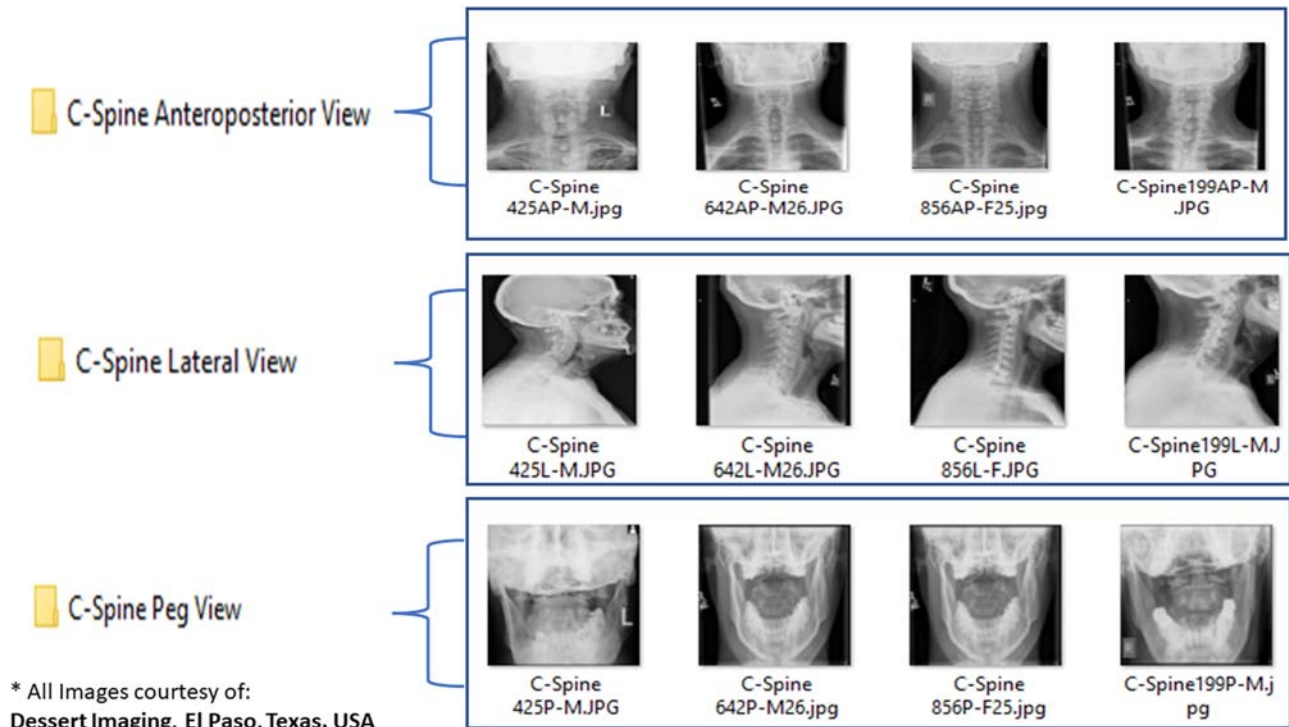


FIGURE 5.23 Image Dataset “X-rays C-Spine” is organized in three folders according with the “Cervical Spine” standard views type. Note: This images dataset is available in the companion directory of the book, in the following directory: “. . .\Exercises_book_ABME\CH5\MATLAB_Build_DCN\X-rays C-Spine.”

5.5.3.6 Procedure

To develop the “custom Deep Convolution Neural Network” using the MATLAB “Deep Network Designer” that has the necessary “AI tools for design custom ANN,” these are organized actually (they are updated frequently adding more options) as follow: “Input,” “Convolutional and Fully Connected,” “Sequence,” “Activation,” “Normalization and Utility,” “Pooling,” “Combination,” “Object Detection,” and “Output,” where:

- “Input” that allow specify layers for loading dataset with the following options: “imageInputLayer,” “image3dInputLayer,” “sequenceInputLayer,” and “roiInputLayer.” Their descriptions are indicated in top section of Fig. 5.24.
- “Convolutional and Fully Connected” that allow integrate “net layer” as: “convolution2dLayer,” “convolution3dLayer,” “groupedConvolution2dLayer,” “transposedConv2dLayer,” “transposedConv3dLayer” and “fullyConnectedLayer.” Their descriptions are indicated in middle section of Fig. 5.24.
- “Sequence” that allows to define “sequence layers” as: “lstmLayer,” “bilstmLayer,” “gruLayer,” “sequence FoldingLayer,” “sequenceUnfoldingLayer,” “flatten

Layer,” and “wordEmbeddingLayer.” Their descriptions are indicated in the lower section of Fig. 5.24.

- “Activation” that allows to define “activation layers” as: “reluLayer,” “leakyReluLayer,” “clippedReluLayer,” “tanhLayer,” “eluLayer,” and “softplusLayer” with their descriptions indicated in top section of Fig. 5.25.
- “Normalization and Utility” that allows to define “normalization types and utilities layers” as: “batchNormalizationLayer,” “crossChannelNormalizationLayer,” “dropoutLayer,” “crop2dLayer,” “crop3dLayer,” “scalingLayer,” “quadraticLayer” with their descriptions indicated in the middle section of Fig. 5.25.
- “Pooling” that allows to define “pooling and unpooling types layers” as: “averagePooling2dLayer,” “averagePooling3dLayer,” “globalAveragePooling2dLayer,” “globalAveragePooling3dLayer,” “maxPooling2dLayer,” “maxUnpooling2dLayer,” “maxPooling3dLayer,” “globalMaxPooling2dLayer,” and “globalMaxPooling3dLayer” with their indicated in bottom section of Fig. 5.25.
- “Combination” that allows to define “different combination layers” as: “additionLayer,” “depth

MATLAB Deep Network Designer layers menu part 1 of 3


















Input	
	imageInputLayer An image input layer inputs 2-D images to a network and applies data normalization.
	image3dInputLayer A 3-D image input layer inputs 3-D images or volumes to a network and applies data normalization.
	sequenceInputLayer A sequence input layer inputs sequence data to a network.
	roiInputLayer (Computer Vision Toolbox™) An ROI input layer inputs images to a Fast R-CNN object detection network.
Convolution and Fully Connected	
	convolution2dLayer A 2-D convolutional layer applies sliding convolutional filters to the input.
	convolution3dLayer A 3-D convolutional layer applies sliding cuboidal convolution filters to three-dimensional input.
	groupedConvolution2dLayer A 2-D grouped convolutional layer separates the input channels into groups and applies sliding convolutional filters. Use grouped convolutional layers for channel-wise separable (also known as depth-wise separable) convolution.
	transposedConv2dLayer A transposed 2-D convolution layer upsamples feature maps.
	transposedConv3dLayer A transposed 3-D convolution layer upsamples three-dimensional feature maps.
	fullyConnectedLayer A fully connected layer multiplies the input by a weight matrix and then adds a bias vector.
Sequence	
	lstmLayer An LSTM layer learns long-term dependencies between time steps in time series and sequence data.
	biLstmLayer A bidirectional LSTM (BiLSTM) layer learns bidirectional long-term dependencies between time steps of time series or sequence data. These dependencies can be useful when you want the network to learn from the complete time series at each time step.
	gruLayer A GRU layer learns dependencies between time steps in time series and sequence data.
	sequenceFoldingLayer A sequence folding layer converts a batch of image sequences to a batch of images. Use a sequence folding layer to perform convolution operations on time steps of image sequences independently.
	sequenceUnfoldingLayer A sequence unfolding layer restores the sequence structure of the input data after sequence folding.
	flattenLayer A flatten layer collapses the spatial dimensions of the input into the channel dimension.
	wordEmbeddingLayer (Text Analytics Toolbox™) A word embedding layer maps word indices to vectors.

FIGURE 5.24 MATLAB Deep Network Designer layers menu part 1 of 3 with layers for: “Input,” “Convolution and Fully Connected,” and “Sequence.”























Activation	
	reluLayer A ReLU layer performs a threshold operation to each element of the input, where any value less than zero is set to zero.
	leakyReluLayer A leaky ReLU layer performs a threshold operation, where any input value less than zero is multiplied by a fixed scalar.
	clippedReluLayer A clipped ReLU layer performs a threshold operation, where any input value less than zero is set to zero and any value above the clipping ceiling is set to that clipping ceiling.
	tanhLayer A hyperbolic tangent (tanh) activation layer applies the tanh function on the layer inputs.
	eluLayer An ELU activation layer performs the identity operation on positive inputs and an exponential nonlinearity on negative inputs.
	softplusLayer It is a deep neural network layer that implements the softplus activation $Y = \log(1 + e^X)$, then output is always positive.
Normalization and Utility	
	batchNormalizationLayer A batch normalization layer normalizes each input channel across a mini-batch. To speed up training
	crossChannelNormalizationLayer channel-wise local response (cross-channel) normalization layer carries out channel-wise normalization
	dropoutLayer A dropout layer randomly sets input elements to zero with a given probability.
	crop2dLayer A 2-D crop layer applies 2-D cropping to the input.
	crop3dLayer A 3-D crop layer crops a 3-D volume to the size of the input feature map.
	scalingLayer It linearly scales and biases an input array U, giving an output $Y = \text{Scale} \cdot U + \text{Bias}$.
	quadraticLayer It takes an input vector and outputs a vector of quadratic monomials constructed from the input elements
Pooling	
	averagePooling2dLayer It performs down-sampling dividing input into rectangular pooling regions and computing avg values each region.
	averagePooling3dLayer It performs down-sampling by dividing input into cuboidal pooling regions and computing avg values each region.
	globalAveragePooling2dLayer It performs down-sampling by computing the mean of the height and width dimensions of the input.
	globalAveragePooling3dLayer It performs down-sampling by computing the mean of the height, width, and depth dimensions of input.
	maxPooling2dLayer It performs down-sampling by dividing the input into rectangular pooling regions, and computing max of each region.
	maxUnpooling2dLayer A max unpooling layer unpoools the output of a max pooling layer.
	maxPooling3dLayer It performs down-sampling dividing three-dimensional input into cuboidal pooling regions & compute max each region
	globalMaxPooling2dLayer It performs down-sampling by computing the maximum of the height and width dimensions of the input.
	globalMaxPooling3dLayer It performs down-sampling by computing the maximum of height, width, and depth dimensions of the input.

FIGURE 5.25 MATLAB Deep Network Designer layers menu part 2 of 3 with layers for: “Activation,” “Normalization and Utility,” and “Pooling.”

ConcatenationLayer,” and “*concatenation Layer*” with their descriptions indicated in figure at the top section of Fig. 5.26.

- “*Object Detection*” that allows to define “*object detection method layers*” as: “*regionProposalLayer*,” “*yolov2ReorgLayer*,” “*yolovTransformLayer*,” “*anchorBoxLayer*” and “*ssdMergeLayer*” with their descriptions indicated in middle section of Fig. 5.26.
- “*Output*” that allows to define “*different output types layers*” as: “*softmaxLayer*,” “*classificationLayer*,” “*regressionLayer*,” “*rpnSoftmaxLayer*,” “*rcnnBox*

RegressionLayer,” “*rpnClassificationLayer*,” “*pixel ClassificationLayer*,” “*dicePixelClassificationLayer*,” “*yolov2OutputLayer*” and “*focalLossLayer*” with their descriptions indicated at the bottom section of Fig. 5.26.

The steps to obtain an “*AI model*” for a “*custom Deep Convolutional Neural Network (DCN)*,” train it, validate it and deploy it through the MATLAB Deep Learning Toolbox” using “*Deep Network Designer*” are summarized in *Table of slides 5.7*, where each step of the example is visually explained using screen sequences with easy to follow instructions.



















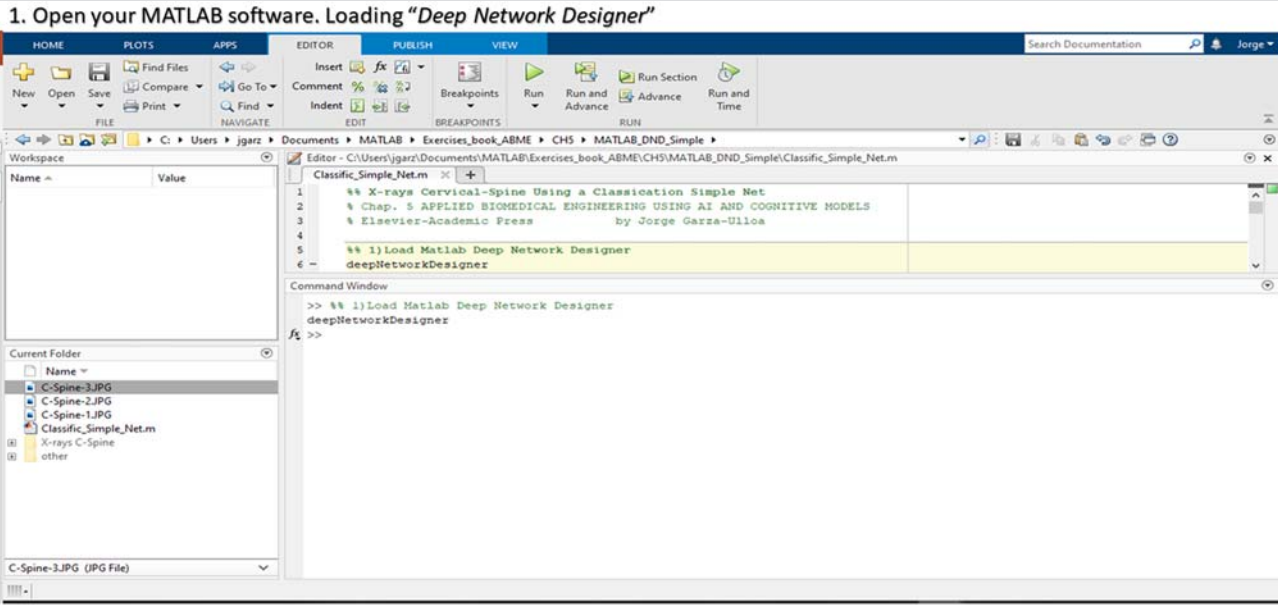
Combination	
	additionLayer An addition layer adds inputs from multiple neural network layers element-wise.
	depthConcatenationLayer A depth concatenation layer takes inputs that have the same height and width and concatenates them along the third dimension (the channel dimension).
	concatenationLayer A concatenation layer takes inputs and concatenates them along a specified dimension. The inputs must have the same size in all dimensions except the concatenation dimension.
Object Detection	
	regionProposalLayer A region proposal layer outputs bounding boxes around potential objects in an image as part of the region proposal network (RPN) within Faster R-CNN.
	yolov2ReorgLayer it create reorganization layer for YOLO v2 object detection network.
	yolov2TransformLayer It represents the transform layer for you look only once version 2 (YOLO v2) object detection network.
	anchorBoxLayer An anchor box layer stores anchor boxes for a feature map used in object detection networks.
	ssdMergeLayer An SSD merge layer merges the outputs of feature maps for subsequent regression and classification loss
Output	
	softmaxLayer A softmax layer applies a softmax function to the input.
	classificationLayer A classic layer computes the cross-entropy loss for multi-class classification problems with mutually exclusive classes.
	regressionLayer A regression layer computes the half-mean-squared-error loss for regression problems.
	rpnSoftmaxLayer A region proposal network (RPN) softmax layer applies a softmax activation function to the input. Use this layer to create a Faster R-CNN object detection network.
	rcnnBoxRegressionLayer A box regression layer refines bounding box locations by using a smooth L1 loss function. Use this layer to create a Fast or Faster R-CNN object detection network.
	rpnClassificationLayer A region proposal network (RPN) classification layer classifies image regions as either object or background by using a cross entropy loss function. Use this layer to create a Faster R-CNN object detection network.
	pixelClassificationLayer A pixel classification layer provides a categorical label for each image pixel or voxel.
	dicePixelClassificationLayer A Dice pixel classification layer provides categorical label for each image pixel/voxel using generalized Dice loss.
	yolov2OutputLayer It creates an output layer for YOLO v2 object detection network
	focalLossLayer A focal loss layer predicts object classes using focal loss.

FIGURE 5.26 MATLAB Deep Network Designer layers menu part 3 of 3 with layers for: “*Combination*,” “*Object Detection*,” and “*Output*.”

Table of slides 5.7 MATLAB Deep Learning Toolbox using “Deep Network Designer” to create a “custom Deep Convolutional Neural Network” to obtain an AI model to “classify Cervical X-rays view types.”

Slide	Description	Screen figure
1	Open your MATLAB software. Loading “Deep Network Designer” Go to the directory “... \Exercises_book_ABME\CH5\ MATLAB_Build_DCN.” Open the script “Classific_Simple_Net.m.” Copy and paste “step 1) Load MATLAB Deep Network Designer” in the command prompt to load the: “deepNetworkDesigner”	 <p>The screenshot shows the MATLAB R2020a interface. The title bar reads "1. Open your MATLAB software. Loading 'Deep Network Designer'". The editor window displays a script named "Classific_Simple_Net.m" with the following content:</p> <pre>1 %% X-rays Cervical-Spine Using a Classification Simple Net 2 % Chap. 5 APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS 3 % Elsevier-Academic Press by Jorge Garza-Ulloa 4 5 %% 1) Load Matlab Deep Network Designer 6 deepNetworkDesigner</pre> <p>The Command Window shows the execution of the script:</p> <pre>>> %% 1) Load Matlab Deep Network Designer deepNetworkDesigner fx >></pre> <p>The current folder is set to "C:\Users\jgarz\Documents\MATLAB\Exercises_book_ABME\CH5\MATLAB_Build_DCN". The file explorer shows files including "C-Spine-3.JPG", "C-Spine-2.JPG", "C-Spine-1.JPG", "Classific_Simple_Net.m", "X-rays C-Spine", and "other".</p>

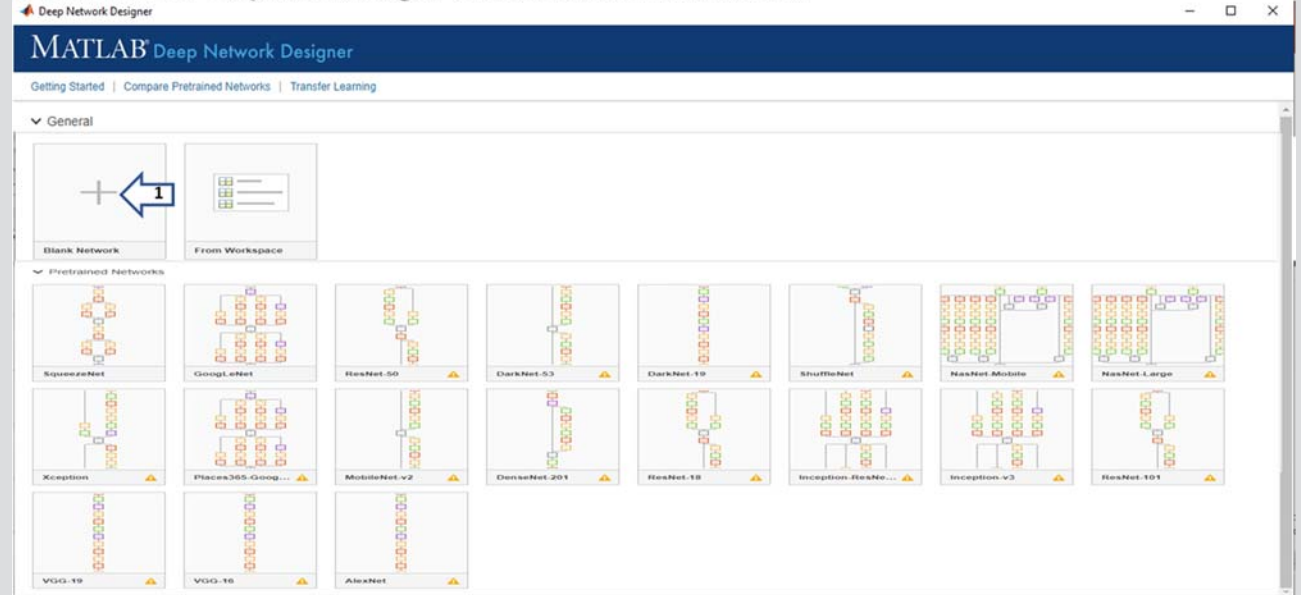
Go to the directory “... \Exercises_book_ABME\CH5\ MATLAB_Build_DCN”. Open the script “Classific_Simple_Net.m”. Copy and paste “step 1) Load MATLAB Deep Network Designer” in the command prompt to load the: “deepNetworkDesigner”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

2

In the “MATLAB Deep Network Designer”: Define a “General > Blank Network”
Click in the “General network: Blank Network” as indicated

2. In the “MATLAB™ Deep Network Designer”: Define a “General > Blank Network”



Click in the “General network: Blank Network” as indicated.

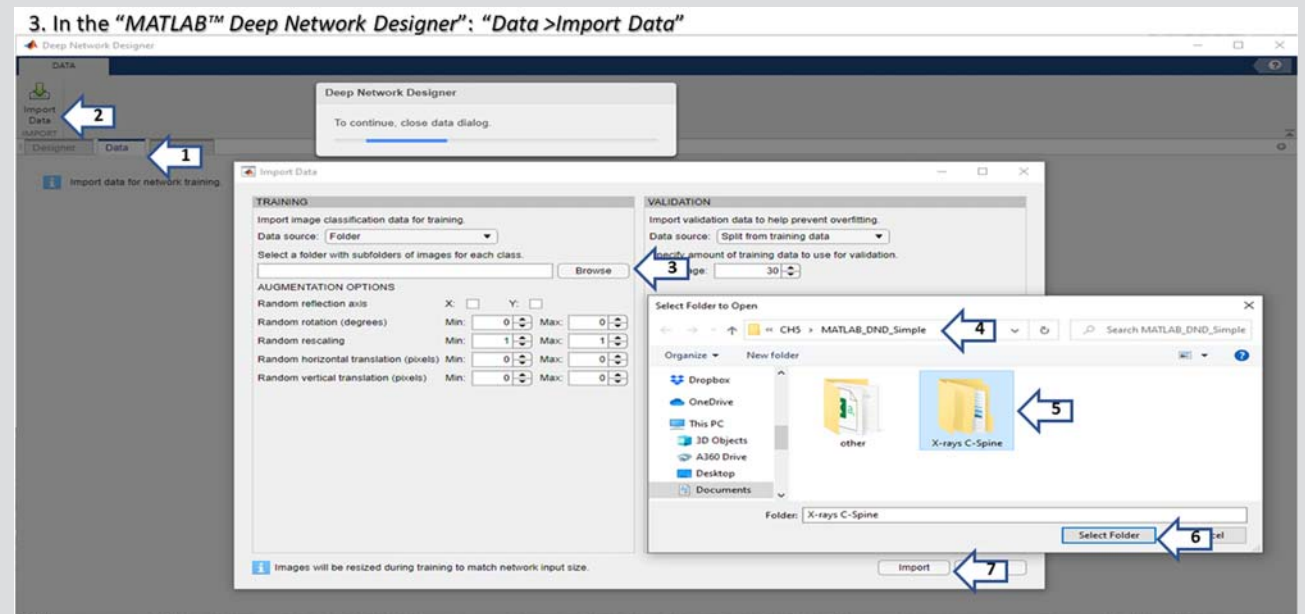
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 3 Description In the "MATLAB Deep Network Designer": "Data > Import Data" In "MATLAB Deep Network Designer" select the "Data" tab, then click on "Import Data" icon, select on "TRAINING: Data source = Folder," click on the "Browse" button and select the folder "X-rays C-Spine" and finally click the "Import" button. Note: Leave Validation = 30%

Screen figure



In "MATLAB™ Deep Network Designer" select the "Data" tab, then click on "Import Data" icon, select on "TRAINING: Data source = Folder", click on the "Browse" button and select the folder "X-rays C-Spine" and finally click the "Import" button. Note: Leave Validation=30%.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

4

In the "MATLAB Deep Network Designer": "Data > Import Data – Training plot"

A plot is generated for the selected "training images classes versus number of them in each class." They are a total of 9 images (observations), in three classes: "C-Spine Anteroposterior View," "C-Spine Lateral View" and "C-Spine Peg View" with three images each



A plot is generated for the selected "training images classes vs number of them in each class". They are a total of 9 images (observations), in three classes: "C-Spine Anteroposterior View", "C-Spine Lateral View" and "C-Spine Peg View" with 3 images each.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

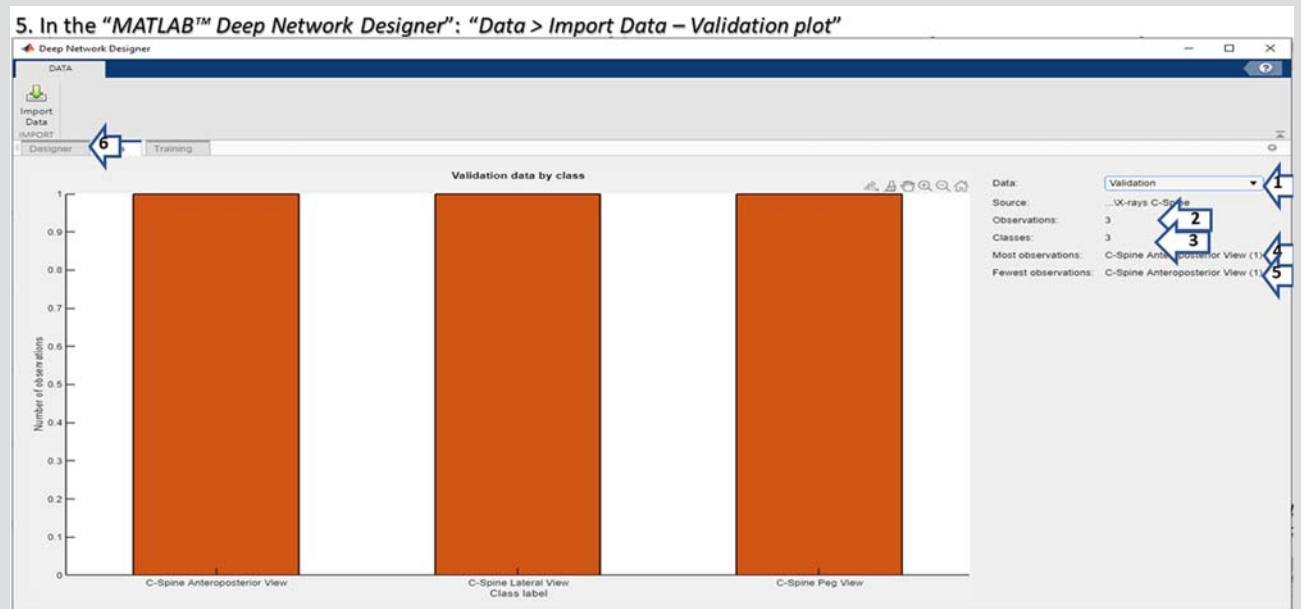
(Continued)

(Continued)

Slide **Description**

5 In the “MATLAB Deep Network Designer”: “Data > Import Data – Validation plot”
A plot is generated for the selected “validation images classes versus number of them in each class.” They are a total of three images for validation (30% of observations), in three classes with 1 observation each. Finally, select the “Designer” tab

Screen figure

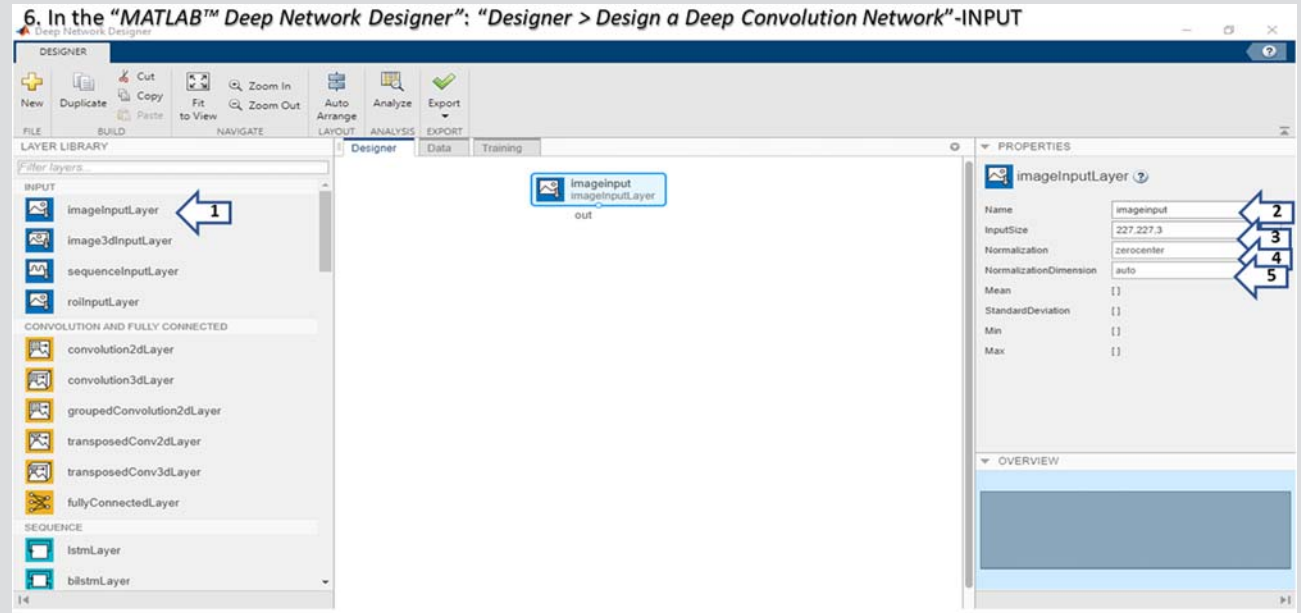


A plot is generated for the selected “validation images classes vs number of them in each class”. They are a total of 3 images for validation (30% of observations), in three classes with 1 observation each. Finally, select the “Designer” tab.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

6

In the “MATLAB Deep Network Designer”: “Designer > Design a Deep Convolution Network”-INPUT
Click and drag an
“INPUT > ImageInputLayer” as shown
in the picture, verify that:
“Name = imageinput,”
“InputSize = 227,227,3,”
“Normalization = zerocenter” and
“NormalizationDimension = auto”



Click and drag an “INPUT > ImageInputLayer” as shown in the picture, verify that: “Name=imageinput”, “InputSize=227,227,3”, “Normalization=zerocenter” and “NormalizationDimension=auto”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 7
Description
In the "MATLAB Deep Network Designer": "Designer > Design a Deep Convolution Network" - CONVOLUTION AND FULLY CONNECTED > convolution2dLayer," and connect it with the previous selection as shown in the picture, verify that: "Name = conv," "FilterSize = 3,3," "NumFilters = 32," "Stride = 1,1," "DilationFactor = 1,1," "Padding = same" and accept the others default parameters

Screen figure

7. In the "MATLAB™ Deep Network Designer" : "Designer > Design a Deep Convolution Network" - CONVOLUTION

The screenshot shows the MATLAB Deep Network Designer interface. On the left, the 'LAYER LIBRARY' is open, showing the 'CONVOLUTION AND FULLY CONNECTED' section. A blue arrow labeled '1' points to the 'convolution2dLayer' icon. In the center, a network diagram shows an 'imageInputLayer' connected to a 'conv convolution2d...' layer. A blue arrow labeled '3' points to the connection line, and another blue arrow labeled '2' points to the 'conv convolution2d...' layer. On the right, the 'PROPERTIES' panel for the 'convolution2dLayer' is shown. A blue arrow labeled '4' points to the 'Name' field, which contains the value 'conv'. Other visible properties include FilterSize (3,3), NumFilters (32), Stride (1,1), DilationFactor (1,1), and Padding (same).

Click and drag an "CONVOLUTION AND FULLY CONNECTED > convolution2dLayer", and connect it with the previous selection as shown in the picture, verify that: "Name=conv", "FilterSize=3,3", "NumFilters=32", "Stride=1,1", "DilationFactor=1,1", "Padding=same" and accept the others default parameters.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

8

In the "MATLAB Deep Network Designer": "Designer > Design a Deep Convolution Network" – NORMALIZATION
Click and drag an "NORMALIZATION AND UTILITY > batch NormalizationLayer," and connect it with the previous selection as shown in the picture, verify that: "Name = batchnorm," "Epsilon = 0.0001," "OffsetLearnRateFactor = 1," and accept the others default parameters

8. In the "MATLAB™ Deep Network Designer": "Designer > Design a Deep Convolution Network" - NORMALIZATION

The screenshot shows the MATLAB Deep Network Designer interface. On the left, the 'LAYER LIBRARY' is open to the 'NORMALIZATION AND UTILITY' section, where the 'batchNormalizationLayer' is selected with a blue arrow labeled '1'. In the center, a network diagram shows an 'imageinput' layer connected to a 'conv convolution2dL...' layer, which is then connected to a 'batchnorm batchNormaliza...' layer. Blue arrows labeled '2', '3', and '4' point to the 'batchnorm' layer in the diagram. On the right, the 'PROPERTIES' panel for the 'batchNormalizationLayer' is shown. The 'Name' is set to 'batchnorm' (arrow '4'), 'Epsilon' is set to '0.0001' (arrow '5'), and 'OffsetLearnRateFactor' is set to '1' (arrow '6'). Other parameters like 'TrainedMean', 'TrainedVariance', 'Offset', 'Scale', 'OffsetL2Factor', 'ScaleL2Factor', 'OffsetInitializer', and 'ScaleInitializer' are shown with their default values.

Click and drag an "NORMALIZATION AND UTILITY > batch NormalizationLayer", and connect it with the previous selection as shown in the picture, verify that: "Name=batchnorm", "Epsilon=0.0001", "OffsetLearnRateFactor=1", and accept the others default parameters.

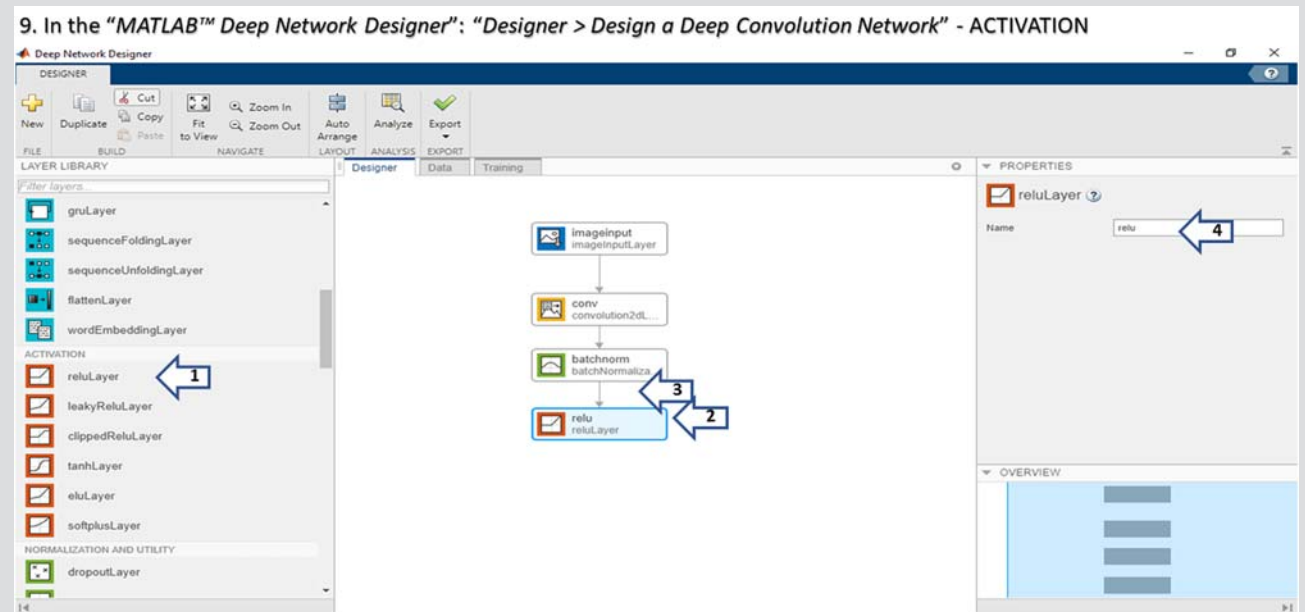
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 9
Description
In the “MATLAB Deep Network Designer”: Designer > Design a Deep Convolution Network –ACTIVATION
Click and drag an “ACTIVATION > reluLayer,” and connect it with the previous selection as shown in the picture, verify that: “Name = relu”

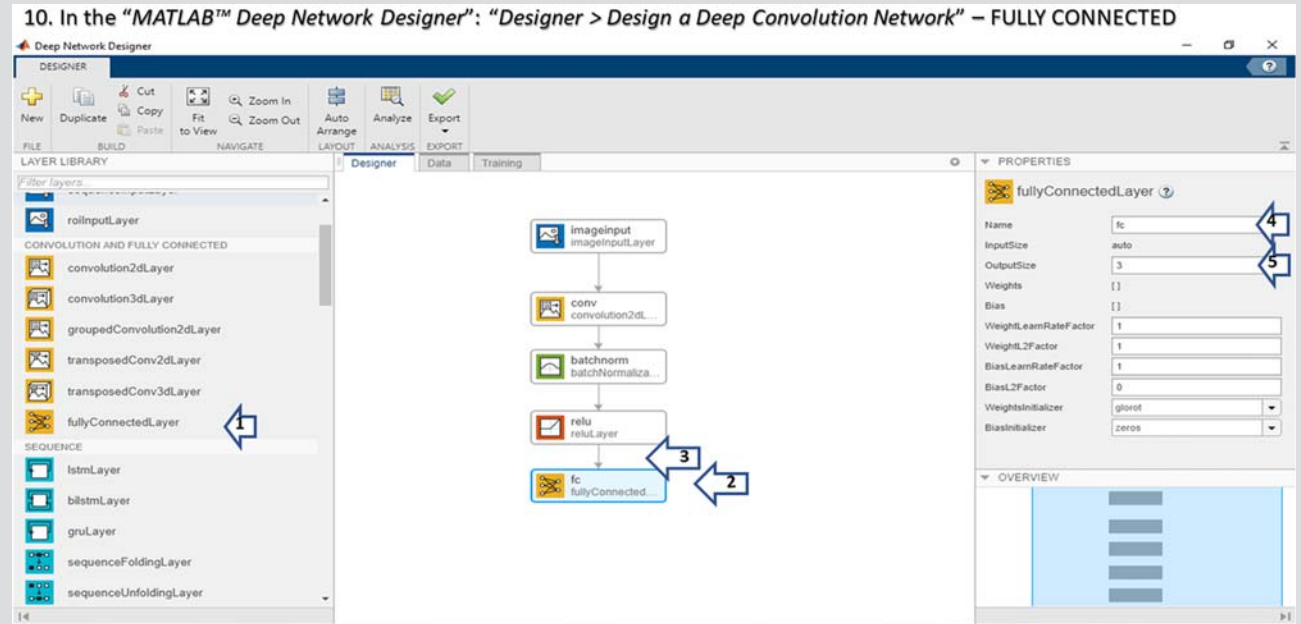
Screen figure



Click and drag an “ACTIVATION > reluLayer”, and connect it with the previous selection as shown in the picture, verify that: “Name=relu”.

10

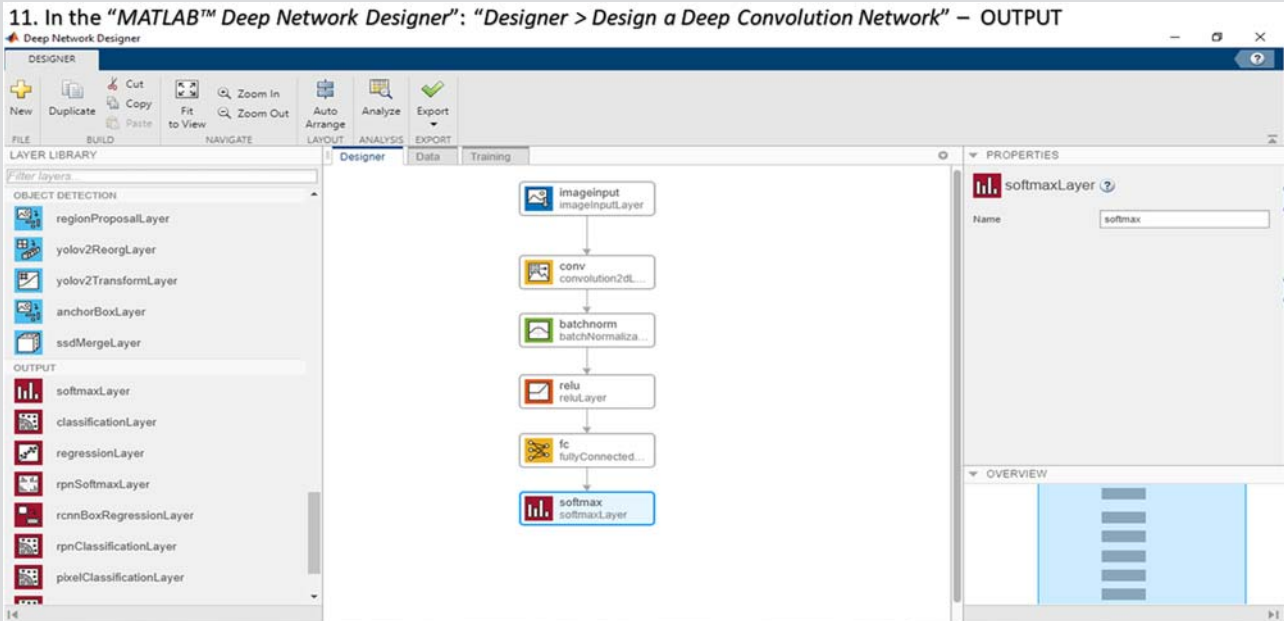
In the “MATLAB Deep Network Designer”: *Designer > Design a Deep Convolution Network* –FULLY CONNECTED
Click and drag an “CONVOLUTION AND FULLY CONNECTED > fullyConnectedLayer,” and connect it with the previous selection as shown in the picture, verify that: “Name = fc,”
“OutputSize = 3” because there are three classes in this image dataset, and accept the others default parameters



Click and drag an “CONVOLUTION AND FULLY CONNECTED > fullyConnectedLayer”, and connect it with the previous selection as shown in the picture, verify that: “Name=fc”, “OutputSize=3” because there are 3 classes in this image dataset, and accept the others default parameters. From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

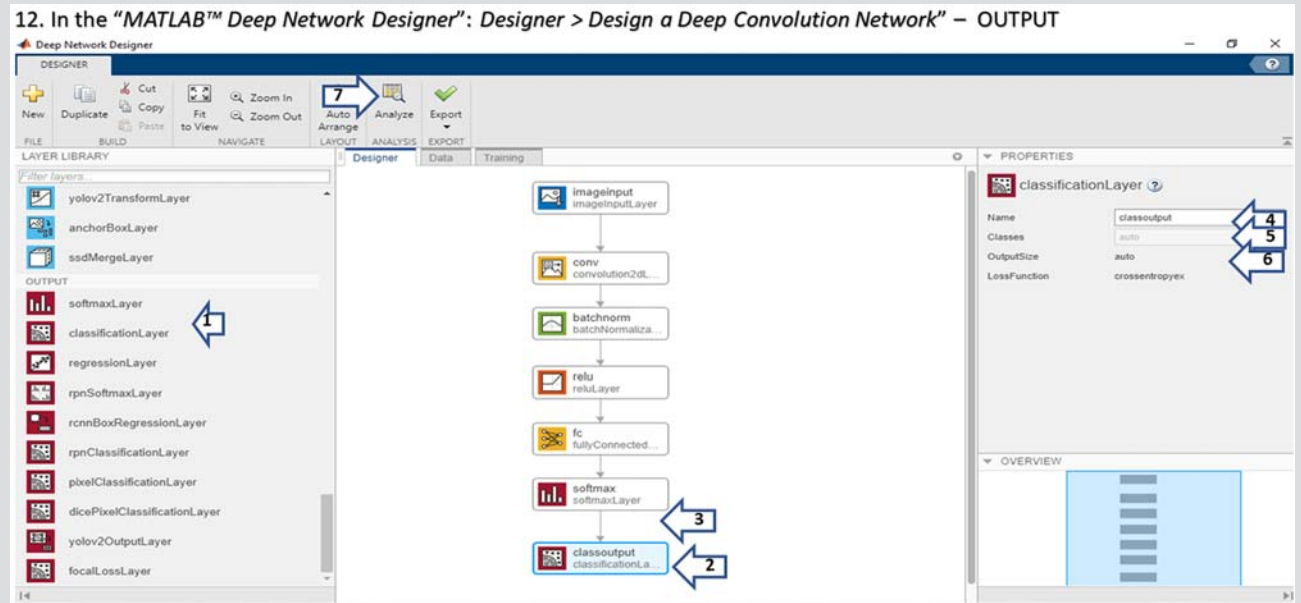
(Continued)

(Continued)

Slide	Description	Screen figure
11	In the "MATLAB Deep Network Designer": Designer > Design a Deep Convolution Network—OUTPUT Click and drag an "OUTPUT > softmax," and connect it with the previous selection as shown in the picture, verify that: "Name = softmax"	<p>11. In the "MATLAB™ Deep Network Designer": "Designer > Design a Deep Convolution Network" – OUTPUT</p>  <p>The screenshot displays the MATLAB Deep Network Designer interface. The central workspace shows a vertical flow of layers: imageInput (imageInputLayer), conv (convolution2dLayer), batchnorm (batchNormalizationLayer), relu (reluLayer), fc (fullyConnectedLayer), and softmax (softmaxLayer). The softmax layer at the bottom is highlighted in blue. On the left, the Layer Library is open to the OUTPUT section, where the softmax layer is selected. On the right, the Properties panel shows the Name property set to 'softmax'. Below the screenshot, a caption reads: "Click and drag an "OUTPUT > softmax", and connect it with the previous selection as shown in the picture, verify that: "Name=softmax"."</p> <p>From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

12

In the “MATLAB Deep Network Designer”: Designer > Design a Deep Convolution Network – OUTPUT
Click and drag an “OUTPUT > classificationLayer,” and connect it with the previous selection as shown in the picture, verify that: “Name = classoutput,” “Classes = auto,” “OutputSize = auto,” and accept the other “LossFunction = crossentropyex” as a default parameter. Finally, click on “Analyze” button



Click and drag an “OUTPUT > classificationLayer”, and connect it with the previous selection as shown in the picture, verify that: “Name=classoutput”, “Classes=auto”, “OutputSize=auto”, and accept the other “LossFunction=crossentropyex” as a default parameter. Finally, click on “Analyze” button.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 13 Description In the “MATLAB Deep Network Designer”: Designer > Design a Deep Convolution Network – ANALYZE “Analyze” open the screen shown, click on the first layer observe the activation of “ $227 \times 227 \times 3$,” “conv with $227 \times 227 \times 32$ ” because of the number of filters, “fc with $1 \times 1 \times 3$ ” because of the three classes/categories, and “classoutput” to be determined after the classification is executed. Note: “0 warnings” and “0 errors”

Screen figure

13. In the “MATLAB™ Deep Network Designer”: Designer > Design a Deep Convolution Network – ANALYZE

The screenshot displays the MATLAB Deep Network Designer interface. The main window shows a network diagram with layers: imageinput, conv, batchnorm, relu, fc, softmax, and classoutput. The 'ANALYSIS RESULT' table is visible, showing the following data:

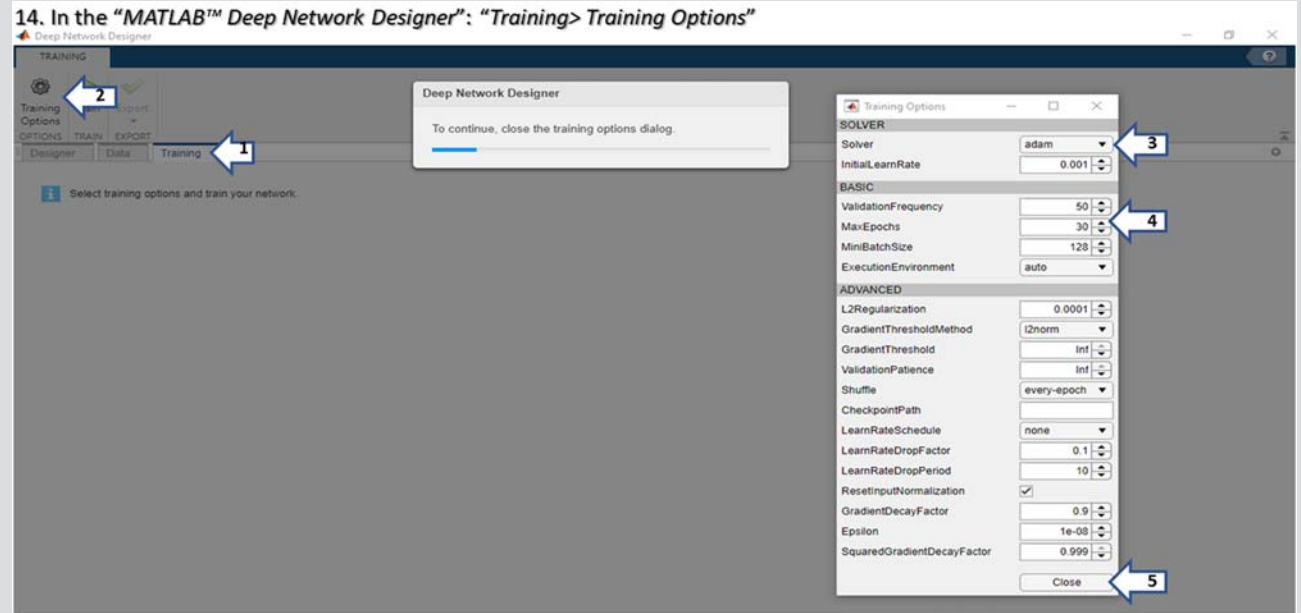
Name	Type	Activations	Learnables
1 imageinput 227x227x3 images with 'zerocenter' normalization	Image Input	227*227*3	
2 conv 32 3x3 convolutions with stride [1 1] and padding 'same'	Convolution	227*227*32	3*3*3=32 1*1=32
3 batchnorm Batch normalization with 32 channels	Batch Normalization	227*227*32	Offset 1*1=32 Scale 1*1=32
4 relu ReLU	ReLU	227*227*32	
5 fc 3 fully connected layer	Fully Connected	1*1*3	Weights 3*1648928 Bias 3*1
6 softmax softmax	Softmax	1*1*3	
7 classoutput crossentropyloss	Classification Output	-	

“Analyze” open the screen shown, click on the first layer observe the activation of “ $227 \times 227 \times 3$,” “conv with $227 \times 227 \times 32$ ” because of the number of filters, “fc with $1 \times 1 \times 3$ ” because of the three classes /categories ,and “classoutput” to be determined after the classification is executed. Note: “0 warnings” and “0 errors”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

14

In the "MATLAB Deep Network Designer": "Training > Train"
Click on "Training" tab, then on "Training Options." At the "Training Options" dialog screen select: "Solver = adam," "MaxEpochs = 30," and leave the other to their default. Finally, click on "Close" button



Click on "Training" tab, then on "Training Options". At the "Training Options" dialog screen select: "Solver = adam", "MaxEpochs = 30", and leave the other to their default. Finally, click on "Close" button.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

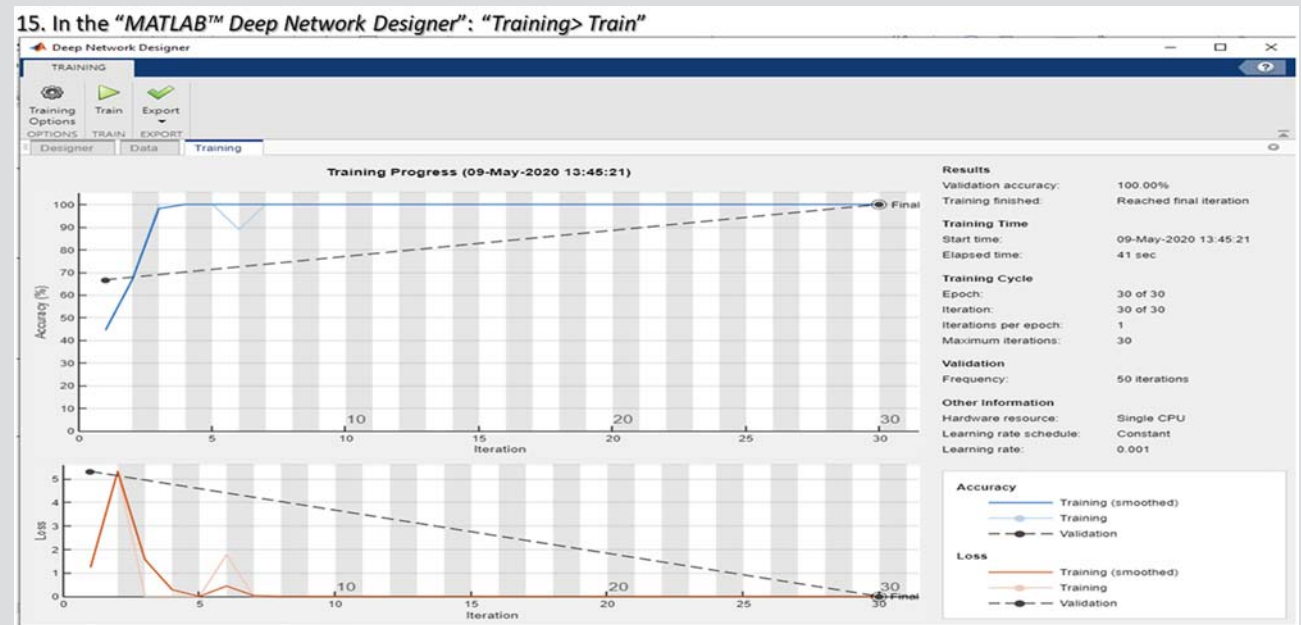
(Continued)

(Continued)

Slide Description

Screen figure

15 In the "MATLAB Deep Network Designer": "Training > Train"
Click on "Train" icon, two plots are generated: "Accuracy (%) versus Iteration" and "Loss versus Iteration."
After some minutes the results are shown with "Validation Accuracy = 100%" and the "Training finished = Reach final iteration."
Note: Retrain if you get lower values for better "Accuracy"



Click on "Train" icon, two plots are generated: "Accuracy (%) vs Iteration" and "Loss vs Iteration". After some minutes the results are shown with "Validation Accuracy=100%" and the "Training finished=Reach final iteration". Note: Re-train if you get lower values for better "Accuracy".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

16

In the "MATLAB Deep Network Designer": "Training > Export"
Click on "Export" button, then on "Export Trained Network and Results."
Press "OK" button at the window confirmation of the "Deep Network Designer"

16. In the "MATLAB™ Deep Network Designer": "Training > Export"

The screenshot displays the MATLAB Deep Network Designer interface during a training session. The top toolbar includes buttons for Training, Train, Export, and Evaluate. A blue arrow labeled '1' points to the 'Export' button. A dropdown menu is open, showing two options: 'Export Trained Network and Results' (with a green checkmark) and 'Generate Code for Training'. A blue arrow labeled '2' points to the 'Export Trained Network and Results' option. A dialog box titled 'Deep Network Designer' is centered on the screen, displaying a green checkmark and the text 'Exported trainedNetwork_1 and trainInfoStruct_1 to the workspace.' A blue arrow labeled '3' points to the 'OK' button in the dialog box. The background shows a training progress plot with 'Accuracy (%)' on the y-axis and 'Iteration' on the x-axis. The plot shows training accuracy increasing from approximately 65% at iteration 0 to 100% at iteration 30. A legend in the bottom right corner identifies the lines as 'Training (smoothed)', 'Training', and 'Validation'. On the right side of the interface, a 'Results' panel provides summary statistics: Validation accuracy: 100.00%, Training finished: Reached final iteration, Training Time: Start time: 09-May-2020 13:45:21, Elapsed time: 41 sec, Training Cycle: Epoch: 30 of 30, Iteration: 30 of 30, Iterations per epoch: 1, Maximum iterations: 30, Validation: Frequency: 50 iterations, Other Information: Hardware resource: Single CPU, Learning rate schedule: Constant, Learning rate: 0.001.

Click on "Export" button, then on "Export Trained Network and Results". Press "OK" button at the window confirmation of the "Deep Network Designer"

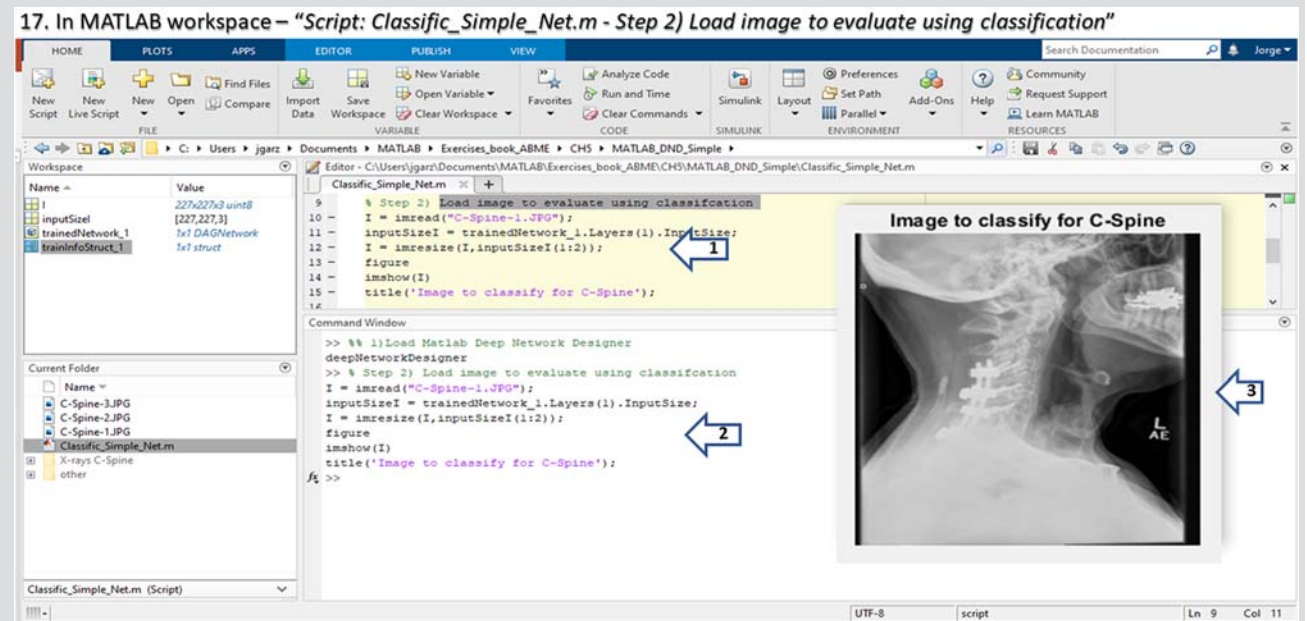
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 17 Description In MATLAB workspace—“Script: *Classific_Simple_Net.m-Step 2) Load image to evaluate using classification*” Verify that the Workspace has two variables: “*trained_Network1*,” and “*trainInfoStruct_1*.” Copy and paste the step 2) Load image to evaluate using classification, and a figure with the image to evaluate is shown. Note: Observe that the “X-rays image is from a patient with surgery back brace implanted”

Screen figure



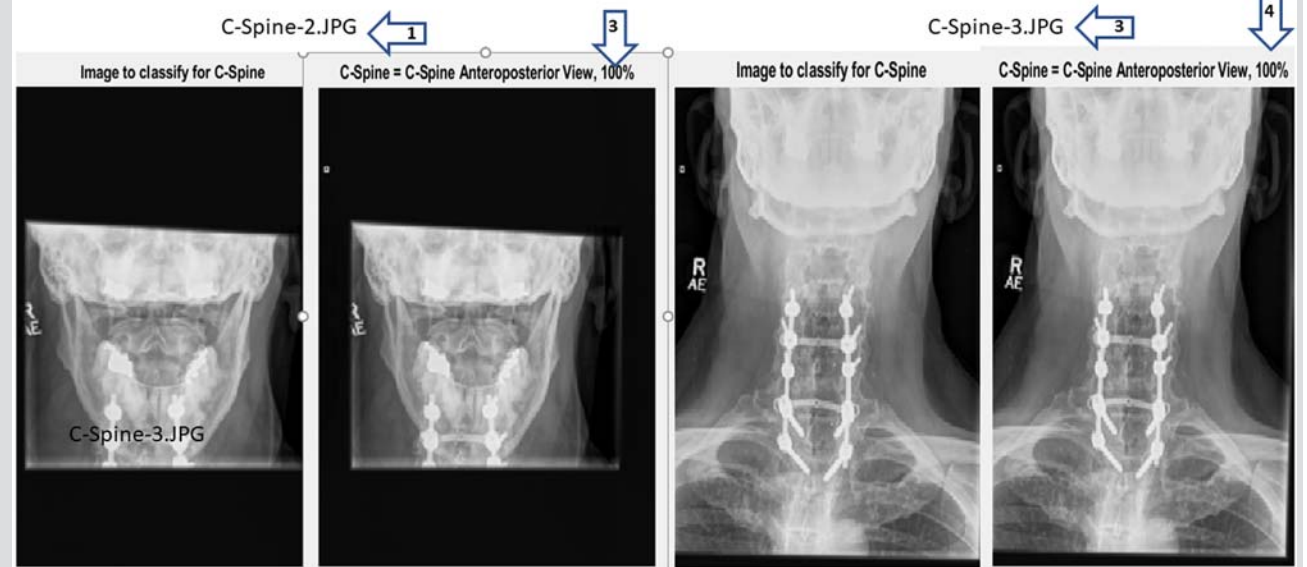
Verify that the Workspace has 2 variables: “*trained_Network1*,” and “*trainInfoStruct_1*”. Copy and paste the step 2) Load image to evaluate using classification, and a figure with the image to evaluate is shown. Note: Observe that the “x-rays is from a patient with surgery back brace implanted”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

In MATLAB workspace—“Script: *Classific_Simple_Net.m-Step 3) Classify the new image*”

Copy and paste the step “3) *Classify the new image,*” and a figure with the image classified and evaluated is shown as a “*C-Spine = C-Spine Anteriorposterior View with 100% of probability*”

19. MATLAB workspace “Script: *Classific_Simple_Net.m - Step 2) and Step 3) for the other C-Spine x-rays in this subdirectory*”



Copy and paste the step “2) and 3) for the other C-Spine in this subdirectory: *C-Spine-2.JPG & C-Spine-3.JPG*”, to classify them showing their probabilities classes = 100%. Finally save all variables to a MAT-file for this in the “Home tab” in the variable section, click “Save Workspace” and close all open figures.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 19 Description
MATLAB workspace “Script: *Classific_Simple_Net.m-Step 2) and Step 3) for the other C-Spine X-rays in this subdirectory*”
Copy and paste the *step “2) and 3) for the other C-Spine in this subdirectory: C-Spine-2.JPG and C-Spine-3.JPG,”* to classify them showing their probabilities classes = 100%. Finally save all variables to a MAT-file for this in the *Home tab* in the variable section, click “*Save Workspace*” and close all open figures

Screen figure

18. In MATLAB workspace – “Script: *Classific_Simple_Net.m - Step 3) Classify the new image*”

```
17 % Step 3) Classify the new image
18 [label,scores] = classify(trainedNetwork_1,I);
19 figure; imshow(I);
20 title("C-Spine = "+string(label) + ", " + ...
21 num2str(100*scores(trainedNetwork_1.Layers(end).Classes == label),3) + "%");
22
```

Command Window

```
>> % Step 3) Classify the new image
[label,scores] = classify(trainedNetwork_1,I);
figure;imshow(I);
title("C-Spine = "+string(label) + ", " + ...
num2str(100*scores(trainedNetwork_1.Layers(end).Classes == label),3) + "%");
fs >>
```

C-Spine = C-Spine Anteroposterior View, 100%

Copy and paste the step “3) Classify the new image”, and a figure with the image classified and evaluated is shown as a “C-Spine = C-Spine Anteriorposterior View with 100% of probability”.

Conclusions

Applying “MATLAB Deep Learning Toolbox” using “Deep Network Designer” tools allows to define, build, train, and deploy an AI “custom Deep Convolutional Neural Network (DCN) model” to “classify cervical X-rays view types.” Type X-Rays image “C-Spine-1.JPG” loaded as indicated in slide 17 and classified in slide 18; is from a patient that had a “surgery back brace implanted,” and the image was classified correctly for the “AI DCN model.”

Recommendations

- Increase the number of “X-rays C-Spine” images in the dataset used on training, validation and test to increase accuracy, defining two groups as “Healthy” and “Cervical affected” patients, to detect “neck pain causes and/or existing trauma” based on “bony structures,” “cartilages,” and “soft tissue (ABCS)” [80].
- Applying “MATLAB Deep Learning Toolbox” using “Deep Network Designer” tool allows to define, build, train, and deploy many “AI neural networks types,” as explained in Chapter 1, Biomedical Engineering and the Evolution of Artificial Intelligence, of this book.

This chapter was based on two “ANN architectural and connection types” plus a general net “Shallow Neural,” and a special kind of learning known as “Transfer Learning from pretrained Deep Learning Networks.” In the next chapter other “ANN” complex architectures with special connections are studied, such as “Recurrent Neural Networks,” “Memory Augmented Neural Networks,” “Modular Neural Networks,” and “Evolutionary Neural Networks” with examples and research on biomedical engineering using existing “AI tools” from “MATLAB” and “IBM Watson Studio.”

References

- [1] Z. Tao, Y. Song, et al., Hybrid collision detection perceptron of the robot in the fusion application, *Fusion. Eng. Des.* 160 (2020) 111800. Available from: <https://doi.org/10.1016/j.fusengdes.2020.111800>. ISSN 0920-3796.
- [2] D.T. Tran, S. Kiranyaz, M. Gabbouj, A. Iosifidis, Progressive operational perceptrons with memory, *Neurocomputing* 379 (2020) 172–181. Available from: <https://doi.org/10.1016/j.neucom.2019.10.079>. ISSN 0925-2312.
- [3] Z. Lv, L. Qiao, Deep belief network and linear perceptron based cognitive computing for collaborative robots, *Appl. Soft Comput.* 92 (2020) 106300. Available from: <https://doi.org/10.1016/j.asoc.2020.106300>. ISSN 1568-4946.
- [4] J.W. Siegel, J. Xu, Approximation rates for neural networks with general activation functions, *Neural Netw.* 128 (2020) 313–321. Available from: <https://doi.org/10.1016/j.neunet.2020.05.019>. ISSN 0893-6080.
- [5] M. Tanaka, Weighted sigmoid gate unit for an activation function of deep neural network, *Pattern Recognit. Lett.* 135 (2020) 354–359. Available from: <https://doi.org/10.1016/j.patrec.2020.05.017>. ISSN 0167-8655.
- [6] J. Hu, H. Tan, C. Zeng, Global exponential stability of delayed complex-valued neural networks with discontinuous activation functions, *Neurocomputing* (2020). Available from: <https://doi.org/10.1016/j.neucom.2020.02.006>. ISSN 0925-2312.
- [7] I. Lorencin, N. Anđelić, J. Španjol, Z. Car, Using multi-layer perceptron with Laplacian edge detector for bladder cancer diagnosis, *Artif. Intell. Med.* 102 (2020) 101746. Available from: <https://doi.org/10.1016/j.artmed.2019.101746>. ISSN 0933-3657.
- [8] K. Bhattacharjee, M. Pant, Hybrid particle swarm optimization-genetic algorithm trained multi-layer perceptron for classification of human glioma from molecular brain neoplasia data, *Cognit. Syst. Res.* 58 (2019) 173–194. Available from: <https://doi.org/10.1016/j.cogsys.2019.06.003>. ISSN 1389-0417.
- [9] M.F. Zorkafli, M.K. Osman, I.S. Isa, F. Ahmad, S.N. Sulaiman, Classification of cervical cancer using hybrid multi-layered perceptron network trained by genetic algorithm, *Procedia Computer Sci.* 163 (2019) 494–501. Available from: <https://doi.org/10.1016/j.procs.2019.12.132>. ISSN 1877-0509.
- [10] V.V. Kamble, R.D. Kokate, Automated diabetic retinopathy detection using radial basis function, *Procedia Computer Sci.* 167 (2020) 799–808. Available from: <https://doi.org/10.1016/j.procs.2020.03.429>. ISSN 1877-0509.
- [11] R. Hrabuska, M. Prauzek, M. Venclikova, J. Konecny, Image reconstruction for electrical impedance tomography: experimental comparison of radial basis neural network and Gauss – Newton method, *IFAC-PapersOnLine* 51 (6) (2018) 438–443. Available from: <https://doi.org/10.1016/j.ifacol.2018.07.114>. ISSN 2405-8963.
- [12] F. Girosi, Some extensions of radial basis functions and their applications in artificial intelligence, *Comput. Math. Appl.* 24 (12) (1992) 61–80. Available from: [https://doi.org/10.1016/0898-1221\(92\)90172-E](https://doi.org/10.1016/0898-1221(92)90172-E). ISSN 0898-1221.
- [13] Available from: <https://www.mathworks.com/help/deeplearning/ug/probabilistic-neural-networks.html> (accessed 25.05.20).
- [14] Available from: <https://www.medicalnewstoday.com/articles/327397#treatment> (accessed 25.05.20).
- [15] Available from: <https://www.webmd.com/cold-and-flu/advanced-reading-types-of-flu-viruses#2> (accessed 22.05.20).
- [16] M. Alweshah, L. Rababa, M.H. Ryalat, A. Al Momani, M.F. Ababneh, African buffalo algorithm: training the probabilistic neural network to solve classification problems, *J. King Saud Univ. Comput. Inf. Sci.* (2020). Available from: <https://doi.org/10.1016/j.jksuci.2020.07.004>. ISSN 1319-1578.
- [17] M. Woźniak, D. Połap, G. Capizzi, G. Lo Sciuto, L. Kośmider, K. Frankiewicz, Small lung nodules detection based on local variance analysis and probabilistic neural network, *Comput. Method. Prog. Biomed.* 161 (2018) 173–180. Available from: <https://doi.org/10.1016/j.cmpb.2018.04.025>. ISSN 0169-2607.
- [18] M.N. Mohanty, H.K. Palo, Child emotion recognition using probabilistic neural network with effective features, *Measurement* 152 (2020) 107369. Available from: <https://doi.org/10.1016/j.measurement.2019.107369>. ISSN 0263-2241.
- [19] J. Garza-Ulloa, Chapter 3 - Kinematic and kinetic measurements of human body, in: J. Garza-Ulloa (Ed.), *Applied Biomechanics*

- using Mathematical Models, Academic Press, 2018, pp. 119–177. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00003-2>. ISBN 9780128125946.
- [20] Available from: <https://my.clevelandclinic.org/health/diseases/7104-diabetes-mellitus-an-overview> (accessed 14.08.20).
- [21] Available from: <https://www.ntu.edu.sg/home/egbhuang/> (accessed 26.05.20).
- [22] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: theory and applications, <https://www.ntu.edu.sg/home/egbhuang/> (accessed 13.08.20).
- [23] Ö.F. Ertuğrul, A novel randomized machine learning approach: reservoir computing extreme learning machine, *Appl. Soft Comput.* 94 (2020) 106433. Available from: <https://doi.org/10.1016/j.asoc.2020.106433>. ISSN 1568-4946.
- [24] Y. Qing, Y. Zeng, Y. Li, G.-B. Huang, Deep and wide feature based extreme learning machine for image classification, *Neurocomputing* 412 (2020) 426–436. Available from: <https://doi.org/10.1016/j.neucom.2020.06.110>. ISSN 0925-2312.
- [25] T. Goel, R. Murugan, Classifier for face recognition based on deep convolutional - optimized Kernel extreme learning machine, *Comput. Electr. Eng.* 85 (2020) 106640. Available from: <https://doi.org/10.1016/j.compeleceng.2020.106640>. ISSN 0045-7906.
- [26] B. Çil, H. Ayyıldız, T. Tuncer, Discrimination of β -thalassemia and iron deficiency anemia through extreme learning machine and regularized extreme learning machine based decision support system, *Med. Hypotheses* 138 (2020) 109611. Available from: <https://doi.org/10.1016/j.mehy.2020.109611>. ISSN 0306-9877.
- [27] R. Johnson, T. Zhang, Effective use of word order for text categorization with convolutional neural networks, 2014, arXiv:1412.1058, <https://arxiv.org/abs/1412.1058>.
- [28] Available from: <https://www.mathworks.com/help/deeplearning/ug/body-fat-estimation.html?jsessionid=a40742999fa0e25beb20cf97193> (accessed 13.08.20).
- [29] C.S. Zuo, Y.-H. Sung, et al., Reduced T2* values in soleus muscle of patients with type 2 diabetes mellitus, Published online November 26, 2012. <https://doi.org/10.1371/journal.pone.0049337> (accessed 14.08.20).
- [30] J. Garza Ulloa, Published Date: 9th June *Applied biomechanics using mathematical models*, Academic Press, 2018.
- [31] K. Jafarian, V. Vahdat, S. Salehi, M. Mobin, Automating detection and localization of myocardial infarction using shallow and end-to-end deep neural networks, *Appl. Soft Comput.* 93 (2020) 106383. Available from: <https://doi.org/10.1016/j.asoc.2020.106383>. ISSN 1568-4946.
- [32] P. Saikia, R.D. Baruah, S.K. Singh, P.K. Chaudhuri, Artificial neural networks in the domain of reservoir characterization: a review from shallow to deep models, *Comput. Geosci.* 135 (2020) 104357. Available from: <https://doi.org/10.1016/j.cageo.2019.104357>. ISSN 0098-3004.
- [33] L. Barazzetti, Point cloud occlusion recovery with shallow feed-forward neural networks, *Adv. Eng. Inform.* 38 (2018) 605–619. Available from: <https://doi.org/10.1016/j.aei.2018.09.007>. ISSN 1474-0346.
- [34] Available from: <http://www.ananth.in/docs/lmtut.pdf#:~:text=The%20Levenberg-Marquardt%20%28LM%29%20algorithm%20is%20the%20most%20widely,to%20provide%20an%20intuitive%20Explanation%20for%20this%20algorithm> (accessed 15.08.20).
- [35] Available from: <https://www.mathworks.com/help/deeplearning/ref/trainbr.html#:~:text=This%20Bayesian%20regularization%20takes%20place%20within%20the%20Levenberg-Marquardt,%2A%20E%20dX%20%3D%20-%20%28jj%2BI%2Amu%29%20%20je> (accessed 15.08.20).
- [36] Available from: <https://medium.com/swlh/backpropagation-step-by-step-13f2b6c0b414> (accessed 15.08.20).
- [37] W. Badr, Auto-Encoder: what Is It? And what is it used for?, <https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726> (accessed 24.07.19).
- [38] Available from: <https://www.mathworks.com/help/deeplearning/ref/trainautoencoder.html> (accessed 14.08.20).
- [39] H. Lu, S. Liu, H. Wei, J. Tu, Multi-kernel fuzzy clustering based on auto-encoder for fMRI functional network, *Expert Syst. Appl.* 159 (2020) 113513. Available from: <https://doi.org/10.1016/j.eswa.2020.113513>. ISSN 0957-4174.
- [40] Y. Zhang, Y. Qiu, Y. Cui, S. Liu, W. Zhang, Predicting drug-drug interactions using multi-modal deep auto-encoders based network embedding and positive-unlabeled learning, *Methods* 179 (2020) 37–46. Available from: <https://doi.org/10.1016/j.ymeth.2020.05.007>. ISSN 1046-2023.
- [41] Z. Ji, Y. Zhao, Y. Pang, X. Li, Cross-modal guidance based auto-encoder for multi-video summarization, *Pattern Recognit. Lett.* 135 (2020) 131–137. Available from: <https://doi.org/10.1016/j.patrec.2020.04.011>. ISSN 0167-8655.
- [42] Y. Ding, L.-P. Tian, X. Lei, B. Liao, F.-X. Wu, Variational graph auto-encoders for miRNA-disease association prediction, *Methods* (2020). Available from: <https://doi.org/10.1016/j.ymeth.2020.08.004>. ISSN 1046-2023.
- [43] J. Li, H. Yan, J. Gao, D. Kong, L. Wang, S. Wang, et al., Matrix-variate variational auto-encoder with applications to image process, *J. Vis. Commun. Image Represent.* 67 (2020) 102750. Available from: <https://doi.org/10.1016/j.jvcir.2019.102750>. ISSN 1047-3203.
- [44] G. Lu, X. Zhao, J. Yin, W. Yang, B. Li, Multi-task learning using variational auto-encoder for sentiment classification, *Pattern Recognit. Lett.* 132 (2020) 115–122. Available from: <https://doi.org/10.1016/j.patrec.2018.06.027>. ISSN 0167-8655.
- [45] D. Monn, Denoising Autoencoders explained, <https://towardsdatascience.com/denoising-autoencoders-explained-dbb82467fc2> (accessed 24.07.19).
- [46] S. Shi, G. Xu, Novel performance prediction model of a biofilm system treating domestic wastewater based on stacked denoising auto-encoders deep learning network, *Chem. Eng. J.* 347 (2018) 280–290. Available from: <https://doi.org/10.1016/j.cej.2018.04.087>. ISSN 1385-8947.
- [47] P. Xiong, H. Wang, M. Liu, S. Zhou, Z. Hou, X. Liu, ECG signal enhancement based on improved denoising auto-encoder, *Eng. Appl. Artif. Intell.* 52 (2016) 194–202. Available from: <https://doi.org/10.1016/j.engappai.2016.02.015>. ISSN 0952-1976.
- [48] M. Nishio, C. Nagashima, S. Hirabayashi, A. Ohnishi, K. Sasaki, T. Sagawa, et al., Convolutional auto-encoder for image denoising of ultra-low-dose CT, *Heliyon* 3 (8) (2017) e00393. Available from: <https://doi.org/10.1016/j.heliyon.2017.e00393>. ISSN 2405-8440.
- [49] Wilkinson, E., Deep learning: sparse autoencoders [online], 2018. Available at: <http://www.ericlewilkinson.com/blog/2014/11/19/deep-learning-sparse-autoencoders> (accessed 18.08.19).

- [50] G. Liu, H. Bao, B. Han, A stacked autoencoder-based deep neural network for achieving gearbox fault diagnosis, *Hindawi*, 2018. Available from: <https://www.hindawi.com/journals/mpe/2018/5105709/> (accessed 28.11.18).
- [51] K. Adem, Diagnosis of breast cancer with Stacked autoencoder and Subspace kNN, *Phys. A Stat. Mech. Appl.* 551 (2020) 124591. Available from: <https://doi.org/10.1016/j.physa.2020.124591>. ISSN 0378-4371.
- [52] L. Che, X. Yang, L. Wang, Text feature extraction based on stacked variational autoencoder, *Microproc. Microsyst.* 76 (2020) 103063. Available from: <https://doi.org/10.1016/j.micpro.2020.103063>. ISSN 0141-9331.
- [53] K. Adem, S. Kiliçarslan, O. Cömert, Classification and diagnosis of cervical cancer with stacked autoencoder and softmax classification, *Expert. Syst. Appl.* 115 (2019) 557–564. Available from: <https://doi.org/10.1016/j.eswa.2018.08.050>. ISSN 0957-4174.
- [54] Available from: <https://www.nationalbreastcancer.org/breast-tumors/> (accessed 18.08.20).
- [55] Available from: <https://www.cancer.org/cancer/breast-cancer/understanding-a-breast-cancer-diagnosis/breast-cancer-grades.html> (accessed 03.06.20).
- [56] Available from: <https://www.cancer.org/cancer/breast-cancer/understanding-a-breast-cancer-diagnosis/stages-of-breast-cancer.html> (accessed 18.08.20).
- [57] W.H. Wolberg, O.L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology, *Proc. Natl Acad. Sci.* 87 (1990) 9193–9196. Available from: <http://www.pnas.org/content/87/23/9193.full.pdf>.
- [58] P.M. Murphy, D.W. Aha, UCI Repository of machine learning databases, University of California, Department of Information and Computer Science, Irvine, CA, 1994. Available from: <http://www.ics.uci.edu/~mlearn/MLRepository.html>.
- [59] Jefkine, Backpropagation in convolutional neural networks, September 2016, <https://jefkine.com/general/2016/09/05/backpropagation-in-convolutional-neural-networks/> (accessed 18.08.20).
- [60] S. Saha, A comprehensive guide to convolutional neural networks—the ELI5 way, December 2018, <https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53> (accessed 18.08.20).
- [61] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, et al., Recent advances in convolutional neural networks, *Pattern Recognit.* 77 (2018) 354–377. Available from: <https://doi.org/10.1016/j.patcog.2017.10.013>. <http://www.sciencedirect.com/science/article/pii/S0031320317304120>. ISSN 0031-3203.
- [62] H. Zhao, J. Zheng, Y. Wang, X. Yuan, Y. Li, Portrait style transfer using deep convolutional neural networks and facial segmentation, *Comput. Electr. Eng.* 85 (2020) 106655. Available from: <https://doi.org/10.1016/j.compeleceng.2020.106655>. ISSN 0045-7906.
- [63] D.K. Atal, M. Singh, Arrhythmia classification with ECG signals based on the optimization-enabled deep convolutional neural network, *Comput. Methods Prog. Biomed.* 196 (2020) 105607. Available from: <https://doi.org/10.1016/j.cmpb.2020.105607>. ISSN 0169-2607.
- [64] M. Zhong, J. LeBien, M. Campos-Cerqueira, R. Dodhia, J.L. Ferres, J.P. Velev, et al., Multispecies bioacoustic classification using transfer learning of deep convolutional neural networks with pseudo-labeling, *Appl. Acoust.* 166 (2020) 107375. Available from: <https://doi.org/10.1016/j.apacoust.2020.107375>. ISSN 0003-682X.
- [65] T.D. Kulkarni, W.F. Whitney, J.B. Tenenbaum, Deep convolutional inverse graphics network, Corpus ID: 14020873, Published in NIPS 2015 Computer Science.
- [66] S. Suresh, S. Mohan, NROI based feature learning for automated tumor stage classification of pulmonary lung nodules using deep convolutional neural networks, *J. King Saud. Univ. Comput. Inf. Sci.* 33 (3) (2019). Available from: <https://doi.org/10.1016/j.jksuci.2019.11.013>. ISSN 1319-1578.
- [67] C.-H. Pham, C. Tor-Díez, H. Meunier, N. Bednarek, R. Fablet, N. Passat, et al., Multiscale brain MRI super-resolution using deep 3D convolutional networks, *Comput. Med. Imaging Graph.* 77 (2019) 101647. Available from: <https://doi.org/10.1016/j.compmedimag.2019.101647>. ISSN 0895-6111.
- [68] G. Piantadosi, M. Sansone, R. Fusco, C. Sansone, Multi-planar 3D breast segmentation in MRI via deep convolutional neural networks, *Artif. Intell. Med.* 103 (2020) 101781. Available from: <https://doi.org/10.1016/j.artmed.2019.101781>. ISSN 0933-3657.
- [69] J. Rocca, Understanding generative adversarial networks (GANs), <https://towardsdatascience.com/understanding-generative-adversarial-networks-gans-cd6e4651a29>, (accessed 25.07.19).
- [70] K.K. Babu, S.R. Dubey, PCSGAN: perceptual cyclic-synthesized generative adversarial networks for thermal and NIR to visible image transformation, *Neurocomputing* 413 (2020) 41–50. Available from: <https://doi.org/10.1016/j.neucom.2020.06.104>. ISSN 0925-2312.
- [71] B. Lei, Z. Xia, F. Jiang, X. Jiang, Z. Ge, Y. Xu, et al., Skin lesion segmentation via generative adversarial networks with dual discriminators, *Med. Image Anal.* 64 (2020) 101716. Available from: <https://doi.org/10.1016/j.media.2020.101716>. ISSN 1361-8415.
- [72] M. Yang, X. Wang, Y. Lu, J. Lv, Y. Shen, C. Li, Plausibility-promoting generative adversarial network for abstractive text summarization with multi-task constraint, *Inf. Sci.* 521 (2020) 46–61. Available from: <https://doi.org/10.1016/j.ins.2020.02.040>. ISSN 0020-0255.
- [73] X. Zhang, S. Ren, J. Sun, K. He, Deep residual learning for image recognition @ Microsoft Research.
- [74] L.H. Shehab, O.M. Fahmy, S.M. Gasser, M.S. El-Mahallawy, An efficient brain tumor image segmentation based on deep residual networks (ResNets), *J. King Saud. Univ. Eng. Sci.* (2020). Available from: <https://doi.org/10.1016/j.jksues.2020.06.001>. ISSN 1018-3639.
- [75] S. Ren, D.K. Jain, K. Guo, T. Xu, T. Chi, Towards efficient medical lesion image super-resolution based on deep residual networks, *Signal Process. Image Commun.* 75 (2019) 1–10. Available from: <https://doi.org/10.1016/j.image.2019.03.008>. ISSN 0923-5965.
- [76] A. Khanna, N.D. Londhe, S. Gupta, A. Semwal, A deep residual U-Net convolutional neural network for automated lung segmentation in computed tomography images, *Biocybern. Biomed. Eng.* (2020). Available from: <https://doi.org/10.1016/j.bbe.2020.07.007>. ISSN 0208-5216.
- [77] F. Deng, Radswiki <https://radiopaedia.org/articles/mammography-views> (accessed 05.05.20).
- [78] J. Garza-Ulloa, Chapter 6 - Application of mathematical models in biomechanics: artificial intelligence and time-frequency analysis, in: J. Garza-Ulloa (Ed.), *Applied Biomechanics using*

- Mathematical Models, Academic Press, 2018, pp. 373–524. ISBN 9780128125946. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00006-8>.
- [79] Available from: <https://healthengine.com.au/info/cervical-spine-x-ray> (accessed 11.05.20).
- [80] Available from: <https://trauma.reach.vic.gov.au/guidelines/imaging-in-trauma/introduction> (accessed 11.05.20).
- [81] Nikos K. Logothetis, Brian A. Wandell, Interpreting the BOLD signal, *Annu Rev Physiol* 66 (2004) 735–769. Available from: <https://doi.org/10.1146/annurev.physiol.66.082602.092845>.

Deep Learning Models Evolution Applied to Biomedical Engineering

6.1 Deep learning models evolution

In the last chapter we studied the underlying principle of “Deep Learning” as the compositional nature of “neural networks” inspired by the biological elements that form the “human brain,” as a collection of “nodes” emulating “brain neurons” and their “neuron synapses” connections as primary elements based on combined primary elements, and their connections to form midlevel elements identified as “Artificial Neural Networks (ANN)” such as: “Feed Forward Neural Network,” “Backpropagation Neural Networks,” “Shallow neural network,” their applications using “Transfer Learning from Pretrained Deep Learning Network and the design of custom “ANN.”

In this chapter we focus on studying “Deep Learning Models Evolution” that combine midlevel elements with different connections “ANN” types to form more complex networks types such as “Recurrent Neural Networks,” “Memory Augmented Neural Networks,” “Modular Neural Networks,” and “Evolutive Neural Networks* as shown at Fig. 1.10 and Fig. 1.11.”

“Artificial Intelligence” is an exponential technology that is continuously growing and evolving, and it is a rapid accelerating aspect of all our lives. I hope that you currently are thinking in “ways to create or combine AI models to apply in your different biomedical engineering projects”.

6.2 Recurrent neural networks types

Recurrent neural networks imply the use of recurrent or feedback connections between neurons, as shown in Fig. 6.1A. Such networks are “Turing complete,” in the sense that they can learn any function, such as spatial and temporal functions. Some frequently used examples of *Recurrent neural networks* are shown in Chapter 1, figure 1.10, these are: *Recurrent Neural Network (RNN) vanilla*, *Long/short-term memory (LSTM)*, *Gated recurrent unit (GRU) networks*, *Recurrent convolutional neural networks (RCNN)*, *Hopfield Network (HN)*, *Boltzmann*

Machine (BM), *Restricted Boltzmann Machine (RBM)*, *Liquid State Machine (LSM)*, *Echo State Network (ESN)*, *Korhonen Network (KEN)*, and many more.

6.2.1 Recurrent Neural Network vanilla

“*Recurrent Neural Network (RNN) vanilla*” takes the previous “output or hidden” states as “inputs” as indicated in the “*RNN general network*” in Fig. 6.1B. The composite input at time “*t*” has some historical information about the happening at time “ $T < t$.” An “RNN” is practically a network with a loop, as shown in Fig. 6.1A. This means that the “output” at the current time step becomes the input to the next time step, with the characteristics that each element considers not only the current input, but all the preceding elements, as shown at the “*RNN sequence time*” in Fig. 6.1C.

The “*RNN models types*” as shown in Fig. 6.1D, are: “one to one,” “one to many,” “many to one,” “many to many,” and “many to many with shifted time,” where:

- “One to one” is when the “RNN model” has “one input x ” and “one output y ,” this means that “ $T_x = T_y = 1$ ” is defining the “traditional neural network” for many “typical AI applications” as shown at Fig. 6.1D 1).
- “One to many” is when the “RNN model” has “one input x ” and “many outputs y ” in a sequence of time represented as “ $T_x = 1, T_y > 1$ ”, this is frequently used for “AI audio and music generation” as shown at Fig. 6.1D 4).
- “Many to one” is when the “RNN model” has “many inputs x_n ” and “one output y ” in a sequence of time represented as “ $T_x > 1, T_y = 1$ ”, this is frequently used for “AI NLP sentiment generation” as shown at Fig. 6.1D 2).
- “Many to many” is when the “RNN model” has “many inputs x_n ” and “many output y_n ” in a sequence of time represented as “ $T_x = T_y$ ”, this is frequently used for “AI NLP entity recognition” as a way to extract information extraction that seeks to locate and classify named entities in unstructured text into predefined categories as shown at Fig. 6.1D 3).

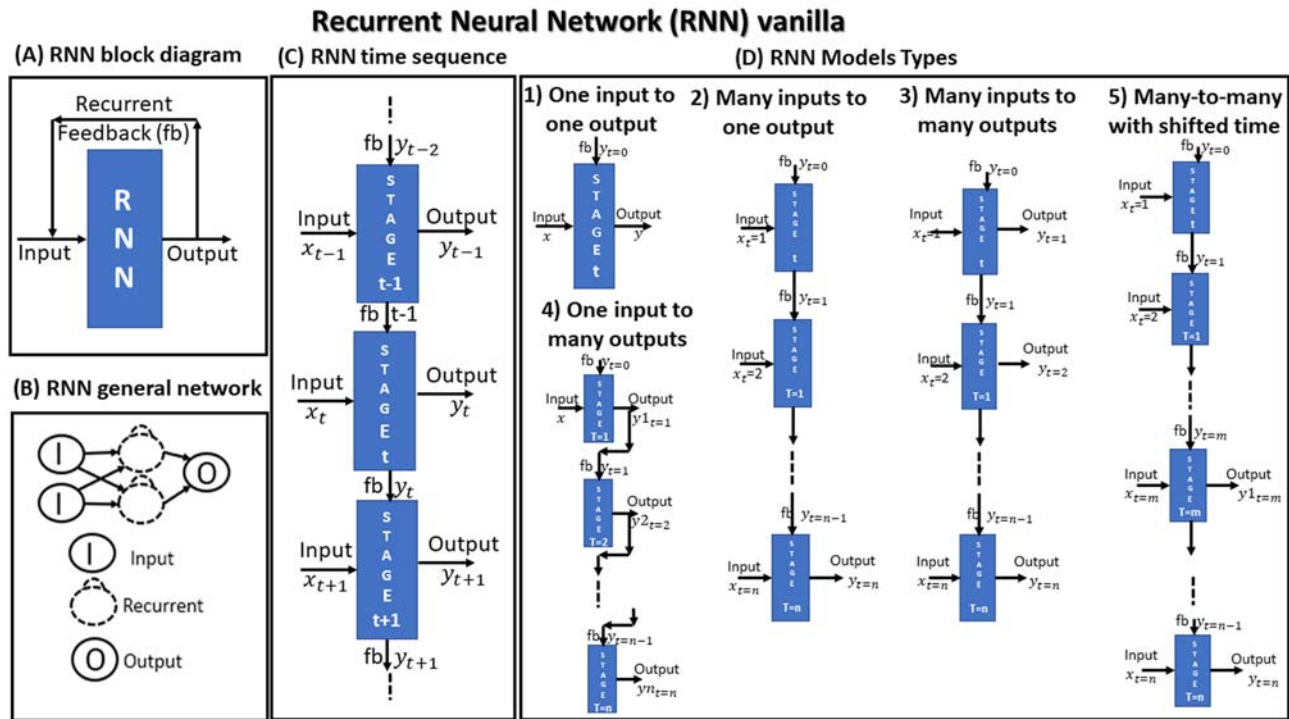


FIGURE 6.1 Recurrent Neural Network (RNN) vanilla: (A) RNN block diagram; (B) RNN general network; (C) RNN sequence time; and (D) RNN Models types.

- “Many to many with shifted time” is when the “RNN model” has “many inputs x_n ” and “many output y_n ” in a shifted sequence of time represented as “ $T_x \neq T_y$,” this is frequently used for “AI NLP Machine translation” as a way to translate text or speech from one language to another as shown at Fig. 6.1D 5).

Some currently used examples of “RNN” in “biomedical engineering” are:

- “Heart sound classification based on improved MFCC features and convolutional recurrent neural networks” by Deng et al. [1]. Based on the heart sound classification, it plays a vital role in the early detection of cardiovascular disorders, especially for small primary health care clinics. In this paper a new heart sound classification method is described based on improved “Mel-frequency cepstrum coefficient (MFCC) features” and “convolutional recurrent neural networks.” The proposed deep learning framework can take advantage of the encoded local characteristics extracted from the “convolutional neural network (CNN)” and the long-term dependencies captured by the “recurrent neural network (RNN).”
- “Multichannel lung sound classification with convolutional recurrent neural networks” by Messner et al. [2] presents an approach for multichannel lung sound classification, exploiting spectral, temporal and spatial

information. They propose a frame-wise classification framework to process full breathing cycles of multi-channel lung sound recordings with a “convolutional recurrent neural network” from a collection of lung sound recordings of lung-healthy subjects and patients with “idiopathic pulmonary fibrosis (IPF).”

- “Classifying sleep–wake stages through recurrent neural networks using pulse oximetry signals” by Casal et al. [3] is on the regulation of the “autonomic nervous system” changes with the sleep stages causing variations in the physiological variables. It classifies the sleep stages into “awake or asleep” using pulse oximeter signals, applying a “recurrent neural network” to heart rate and peripheral oxygen saturation signals to classify the sleep stage every 30 seconds. The network architecture consists of two stacked layers of “bidirectional gated recurrent units (GRUs)” and a “softmax layer” to classify the output.

“RNN” has the following advantages: [4] (1) model size is not increased with size input; (2) input size can be of practically every size needed; (3) computation considers historical information, and others. The main disadvantage is that computation is “slow taking more processing time.” “RNN” models are mostly used in the fields of “Natural Language Processing (NLP)” and “speech recognition.”

6.2.2 Long/short-term memory

“Long/Short-Term Memory (LSTM)” is a special “RNN” capable of learning long-term dependencies simulating in its feedback connections a “general purpose computer.” All “RNN” have the form of a chain of repeating modules that are analyzed as a sequence in time, using a “*tanh*” activation function in a single layer, as indicated in Fig. 6.2A, and connected as repeating modules in time, as indicated in Fig. 6.2D, where each line carries an entire vector from the output of one node to the input of the next one. “LSTM” have also the form of a chain of repeating modules with “LSTM gates,” as indicated in Fig. 6.2B and connected as repeating modules in Fig. 6.2E. “LSTM” has basically three types of memory cells an “Forget Gate” that decides what information is updated or discarded from the cell; an “Input Gate” that decides which values from the input to update the memory state; and an “Output Gate” that decides what to output based on the input and the memory of the cell [5]. These operations are made through “two buses or tracks: B1 and B2” and “four layers: L1, L2, L3, and L4,” as shown in Fig. 6.2B and their equation in Fig. 6.2C, where:

- “B1” is the bus or track that connects the four layers taking the last output “ $B1 = y_{t-1}$,” “B2” is the bus or track taking “the last cell state C_{t-1} ” and delivers at the end the “actual cell state C_t .” It runs straight with two

pointwise: multiplication in “layer L1” and addition in “layer L2.” “B2” at the end is the “actual cell state.”

- “L1” is known as the “forget gate layer” with “sigmoid activation function” that outputs numbers between “zero that indicate forget the actual value” and “one that indicates stay with the actual value,” describing how much of each component should be let through to “B2” to alter the “actual cell state.” The equation for “L1” representing this step is shown in Fig. 6.2C.
- “L2” is known as the “input gate layer” with “sigmoid activation function” that outputs numbers between “zero to don’t store actual value” and “one to store the actual value,” describing how much of each component should store through to “L3” to alter the “actual cell state.”
- “L3” is known as the “update the cell state.” It uses a “*tanh* activation function” very similar to “sigmoid” but with a range from “-1 to +1” controlled by the “input gate” from “L2” to add the new information or not to the “actual cell state.”
- “L4” is known as the “Output Gate.” Here the decision of what is going to be the output is taken. “L4” has in first place an “sigmoid activation function” that decides which parts of the cell state are going to be the output, and finally places the cell state through a “*tanh* activation function” to the output “ y_t .”

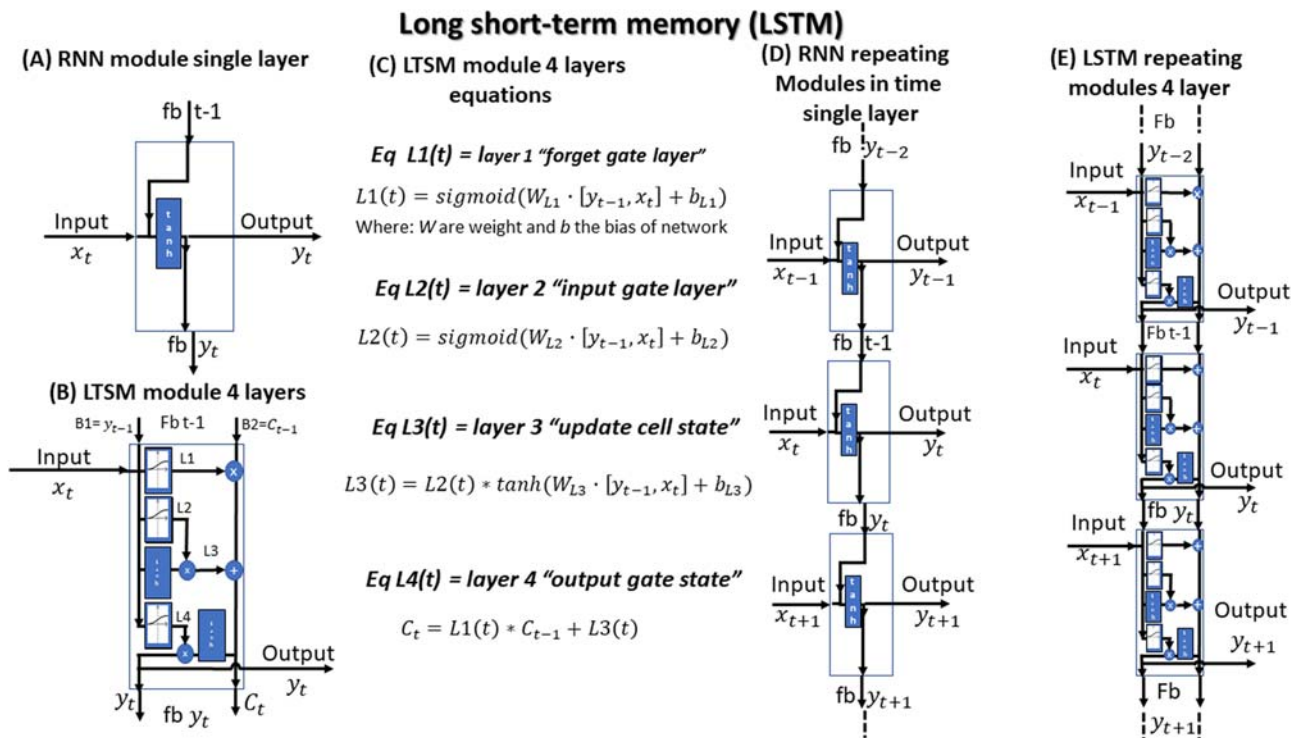


FIGURE 6.2 Long/short-term memory (LSTM): (A) RNN module single layer; (B) LTSM module 4 layers; (C) shows the LTSM module 4 layers equations, (D) RNN repeating modules single layer; and (E) LSTM repeating modules 4 layer.

Some currently examples of “LSTM” in “Biomedical Engineering” are:

- “Deep historical long short-term memory network for action recognition” by Cai et al. [6] Due to the fact that human action recognition technology has received increasing interest recently a deep historical “long/short-term memory network” for video-based tennis action recognition and general action recognition was developed. First, the spatial representations are extracted from each frame using a “pretrained convolutional neural network (CNN).” To describe the temporal information, a stacked multilayer “long/short-term memory network (LSTM)” was used with excellent results.
- “Heart sound segmentation via Duration Long–Short Term Memory neural network” by Chen et al. [7], based on heart sound segmentation, which aims to detect the first and second heart sounds in a “phonocardiogram.” It is an essential step to automatically analyze “heart valve diseases.” In this paper, a “Duration Long–Short-Term Memory network (Duration LSTM)” is proposed as a method to investigate real-world phonocardiogram datasets.
- “Abnormal heart sound detection using temporal quasi-periodic features and long short-term memory without segmentation” by Zhang et al. [8]. They proposed a novel method for “abnormal heart sound detection using temporal quasi-periodic features” and “long/short-term memory without segmentation.” In the proposed method, the “spectrogram of the heart sound signal” is extracted using the “short-time Fourier transform,” then “temporal quasi-periodic features of the heart sound signal” are calculated by the average magnitude difference function from the spectrogram in different frequency bands, and finally they extract the dependency relation within the temporal quasi-periodic features of the method by applying “long/short-term memory.”

“LSTM” can be used for classifying, processing, and predictions, because the “LSTM” process uses single data points as images to sequences of data as text, speech, audio, and video [9]. In summary, “Long/Short-Term Memory (LSTM) networks are a type of recurrent neural network capable of learning order dependence in sequence prediction problems.”

Note: Please see the example: “Research 6.1, section 6.2.3.1 LSTM to Classify videos about human body movements and detect human falls.”

6.2.3 Gated recurrent unit networks

“Gated Recurrent Unit (GRU)” is a variant recurrent neural networks (RNN) like a “long/short-term memory

(LSTM)” unit but “without an output gate” using a gating mechanism. A “GRU” can be considered a specific variation of a “LSTM” unit because both have a similar design and produce equal results in some cases. “GRU” uses a gating mechanism to control and manage the flow of information between cells in the neural network. “GRUs” are able to solve the “vanishing gradient problem*” by using an “update gate” and a “reset gate” as shown in the “GRU module of 3 layers” in Fig. 6.3A, and its equations in Fig. 6.3B, where:

- the “update gate” controls information that flows into memory.
- the “reset gate” controls the information that flows out of memory.
- The “update gate and reset gate” are two vectors that decide which information will get passed on to the output as shown in the sequence on time as a GRU with three layers of repeating modules in Fig. 6.3C. They can be trained to keep information from the past or remove information that is irrelevant to the prediction.

Some current examples of “GRU” in “Biomedical Engineering” are:

- “Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM” by Shahid et al. [10]. Proposed forecast models comprising “autoregressive integrated moving average (ARIMA),” “support vector regression (SVR),” “long/short term memory (LSTM),” “bidirectional long/short-term memory (Bi-LSTM),” and “Gated recurrent unit (GRU) networks” are assessed for time series prediction of confirmed cases, deaths, and recoveries due to COVID-19 in 10 major countries affected. The performance of models is measured by mean absolute error, root mean square error, and r2_score indices. In most cases, the “Bi-LSTM” model outperforms in terms of endorsed indices. The model ranking from good performance to the lowest in entire scenarios are “Bi-LSTM,” “LSTM,” “GRU,” “SVR,” and “ARIMA.”
- “A molecular generative model of ADAM10 inhibitors by using GRU-based deep neural network and transfer learning” by Shi et al. [11]. A “gated-recurrent-unit (GRU)” network combined with transfer learning was successfully employed to establish a molecular generative model of ADAM10 inhibitors. The results showed that the GRU-based generative model can learn accurately the SMILES grammars of the molecules and is capable of generating novel potential ADAM10 inhibitors.
- “Fused GRU with semantic-temporal attention for video captioning” by Gao et al. [71]. A combination of semantic and temporal attention for video captioning with an end-to-end pipeline named “Fused GRU

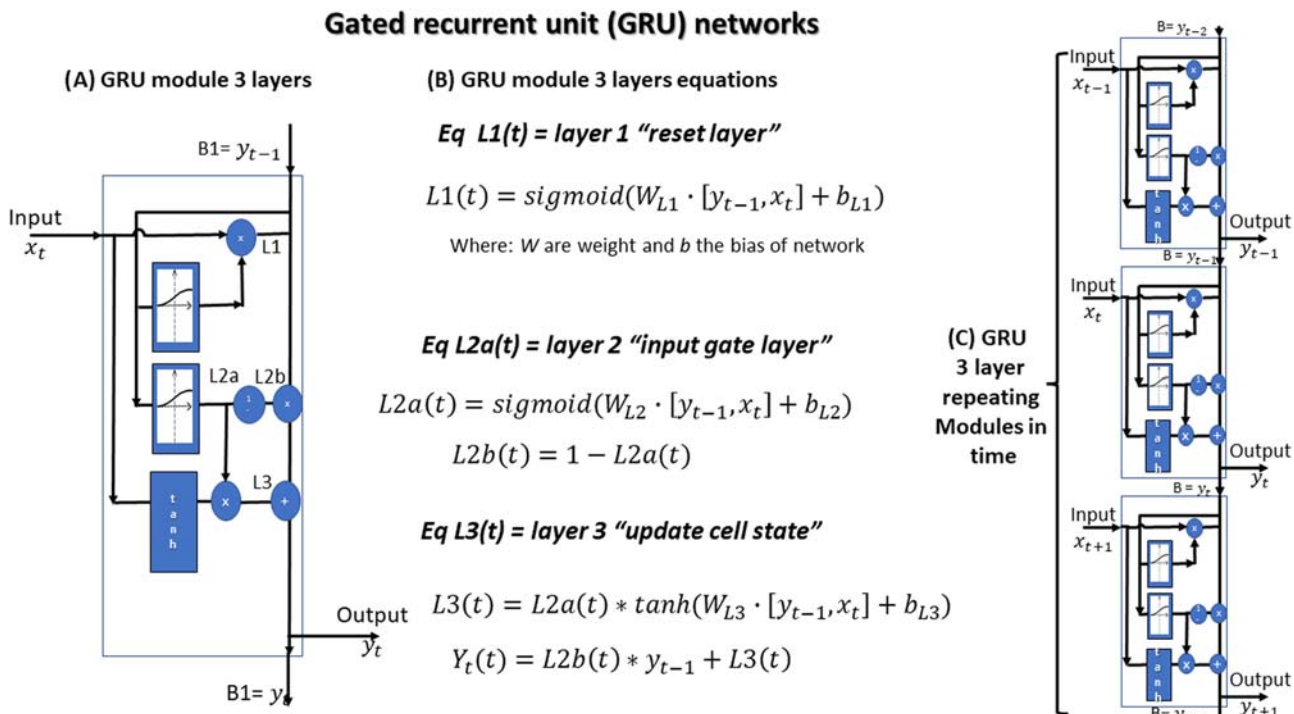


FIGURE 6.3 Gated recurrent unit (GRU) networks: (A) GRU module 3 layers; (B) GRU module 3 layers equations; and (C) GRU repeating modules 3 layers.

with *Semantic-Temporal Attention (STA-FG)*,” can explicitly incorporate the high-level visual concepts for the generation of semantic-temporal attention for video captioning. The encoder network aims to extract visual features from the videos and predict their semantic concepts, specifically, the decoder combines both visual and semantic representation, and incorporates a semantic and temporal attention mechanism in a “fused GRU network” to accurately learn the sentences for video captioning.

Note:* Vanishing gradient problem occurs for the calculated partial derivatives used to compute the gradient as one goes deeper into the network. Since the gradients control how much the network learns during training, if the gradients are small or zero, then little to no training can take place, leading to poor predictive performance.

6.2.3.1 Research 6.1 LSTM to classify videos about human body movements and detect human falls

6.2.3.1.1 Case for research

“Obtain an LSTM model from MATLAB Deep Learning Toolbox using a Deep Network Designer to classify and then identify from videos from body movements as the human falls.”

6.2.3.1.2 General objective

Apply “MATLAB Deep Learning Toolbox” to define, build, train, and deploy an “LSTM Neural Network model to classify from human body movements captured in videos and identify in new videos the kind of human body movements with special emphasis on walking human falls.”

6.2.3.1.3 Specific objectives

- Load video dataset and convert frames to vectors and save sequences.
- Create two partitions for training with 90% and validation 10% of the video dataset and visualize the “Sequence length” from all videos.
- Define, create, visualize, analyze, and train the custom “LSMT NN” using the MATLAB “Deep Network Designer” available in the “MATLAB Deep Learning Toolbox.”
- Export the “LSMT network model” to be integrated in the “pretrained GoogLeNET.”
- Assemble the “Pretrained GoogleNet net and LSTM” for “Video Classification”.
- “Classify using new video data with human body movements to detect as human falls.”

6.2.3.1.4 Background for “Walking Human falls”

“During walking each step is really a tiny fall, and it can be represented by a mathematical model.” Human falls are

possible for many reasons but they have more severe consequences in seniors; a fall can be devastating in old age when the probability of risk of falling is increased along with serious injuries. According to the “*Center for Disease Control and Prevention*” elderly fall statistics [12]:

- One in four seniors experiences a fall every year.
- Every 11 seconds, a senior is treated for a fall in the emergency room.
- Every 19 minutes, a senior dies from a fall.

The main key risk factors for falls in the elderly are:

- Muscle weakness, arthritis, balance, and gait problems.
- Visual impairments.
- Medication-induced sleepiness/dizziness.
- Environmental hazards.
- Chronic conditions, such as:
 - “*Heart disease*” based on their symptoms, such as low blood pressure, heart failure, and arrhythmias that can lead to fainting;
 - “*Brain disease*” such as epilepsy, Parkinson’s disease, Alzheimer’s disease, and other cognitive disorders;
 - “*Diabetes*” that can lead to visual impairments, numbness in the legs, and, in extreme cases, diabetic coma;

- “*Inner ear problems*,” as the most important organ for our sense of balance is located in our inner ear, problems with it can cause vertigo;
- “*Alcoholism*,” especially alcohol abuse paired with certain medications, can easily cause a person to fall;
- And many others.

Falls in the older population are public health and community problems with adverse physical, medical, psychological, social, and economic consequences. Some of the consequences are disability and deformity, curtailment of routine social activities, fear of repeated falls, cost of medical care associated with injuries, and loss of income.

6.2.3.1.5 Dataset

The “*image dataset is a small recompilation of 36 videos in MP4 format*” organized into three categories of human movements as “*fall*,” “*run*,” and “*walk*,” as shown in Fig. 6.4. With the main goal being “*proof of the concept that walking human fall can be detected using videos applying an LSMT AI model*,” so that an alarm signal is raised for personnel to assist the patients falls in care homes, assisted living and senior care, hospitals, etc.

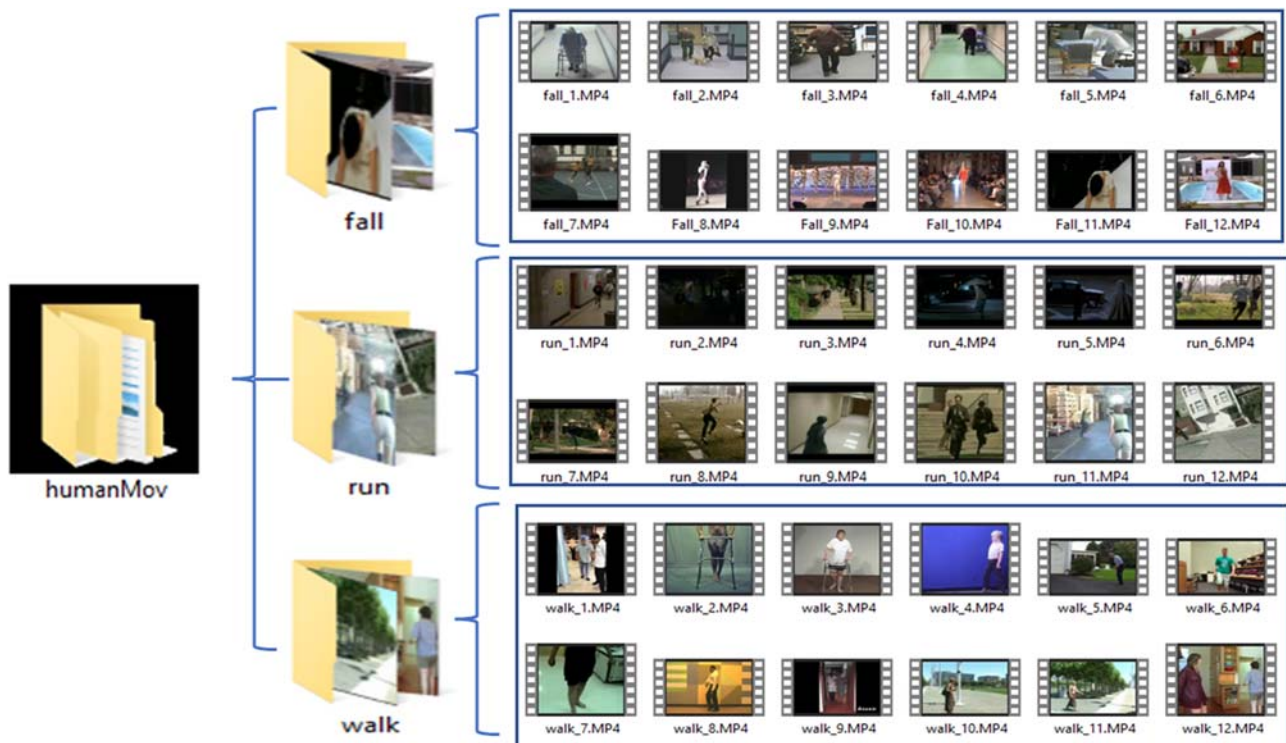


FIGURE 6.4 The Image Dataset “*humanMov*” is organized in three folders according to three basic human body movement videos in “*MP4*” format: “*fall*,” “*run*,” and “*walk*.”

Note: This images dataset is available in the companion directory of the book, in the following folder “*..\Exercises_book_ABMECH6MATLAB_LSTM_videos.*”

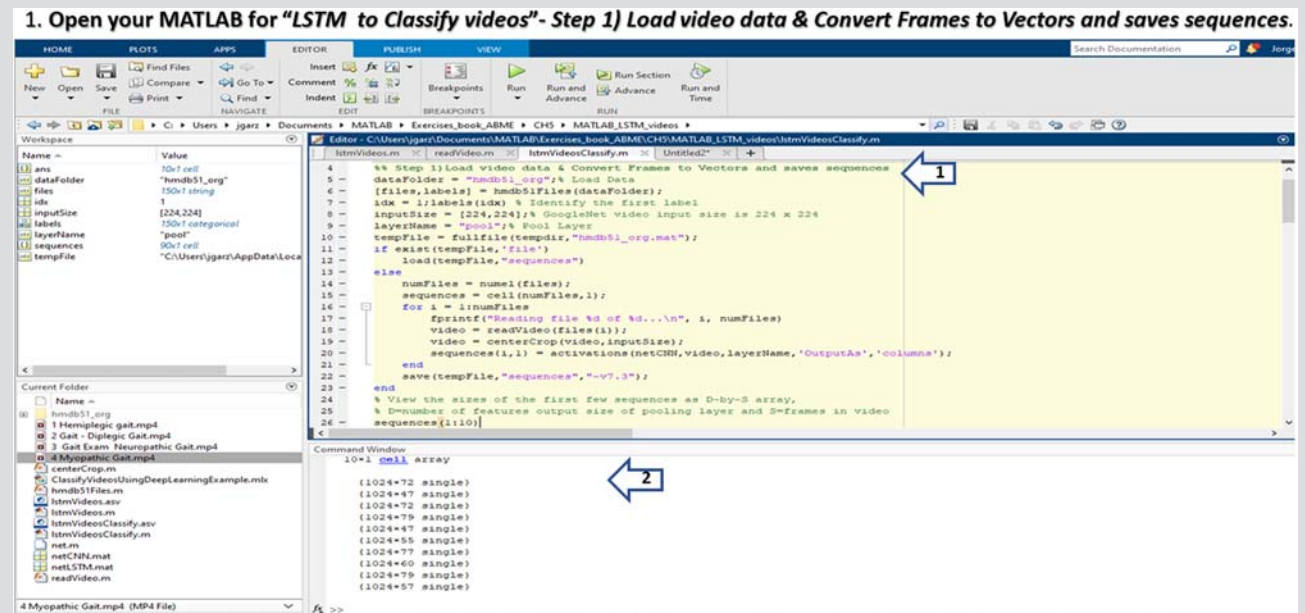
6.2.3.1.6 Procedure

The steps to “*Obtain an LSTM model from MATLAB Deep Learning Toolbox using a Deep Network Designer*

to classify the human body movements on videos and to detect walking human falls” are summarized in *Table of slides 6.1* and each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

Table of slides 6.1 Steps for MATLAB Deep Learning toolbox to create an LSTM NN to classify videos about human body movements and detect walking human falls.

Slide 1
 Description
 Open your MATLAB to "LSTM to Classify videos" —Step 1) Load video data & Convert Frames to Vectors and saves sequences. Go to the directory "...|Exercises_book_ABME\CH6| MATLAB_LSTM_videos." Open the script "IsmtVideosClassify.m." Copy and paste in the command prompt: "Step 1) Load video data & Convert Frames to Vectors and saves sequences." It shows on the screen a sample from the sequences generated from the video data



Go to the directory "...|Exercises_book_ABME\CH6| MATLAB_LSTM_videos". Open the script "IsmtVideosClassify.m". Copy and paste in the command prompt: "Step 1) Load video data & Convert Frames to Vectors and saves sequences". It shows in the screen a sample from the sequences generated from the video data

2

MATLAB for “LSTM to Classify videos”—Step 2) Partition data and visualize sequence lengths. Copy and paste in the command prompt: “2) Partition data and visualize sequence lengths.” It shows in a figure the “Sequence Lengths versus Frequency” generated from the video data. Note: Excessive sequence length will make to take longer to process and be with less accuracy the AI model

2. MATLAB for “LSTM to Classify videos”- Step 2) Partition data and visualize sequence lengths.

```
29 %% Step 2) Partition data and visualize sequence lengths
30 numObservations = numel(sequences);
31 idx = randperm(numObservations);
32 N = floor(0.9 * numObservations); % Train 90 percent Validation 10 percent
33 idxTrain = idx(1:N); sequencesTrain = sequences(idxTrain)
34 idxValidation = idx(N+1:end); sequencesValidation = sequences(idxValidation)
35 %Remove Long Sequences
36 numObservationsTrain = numel(sequencesTrain); sequenceLengths = zeros(1, numObservationsTrain);
37 for i = 1:numObservationsTrain
38     sequence = sequencesTrain(i);
39     sequenceLengths(i) = size(sequence,2);
40 end
41 figure; histogram(sequenceLengths)
42 title("Sequence Lengths"); xlabel("Sequence Length"); ylabel("Frequency");
43 maxLength = 500; % limit length <=500
44 idx = sequenceLengths > maxLength;
45 sequencesTrain(idx) = []; labelsTrain(idx) = [];
```

```
sequenceLengths(i) = size(sequence,2);
end
figure; histogram(sequenceLengths)
title("Sequence Lengths"); xlabel("Sequence Length"); ylabel("Frequency");
maxLength = 500; % limit length <=400
idx = sequenceLengths > maxLength;
sequencesTrain(idx) = []; labelsTrain(idx) = [];
```

Sequence Length	Frequency
0-100	18
100-200	8
200-300	4
300-400	1
400-500	1
500-600	1
600-700	1
700-800	1
800-900	1
900-1000	1

Copy and paste in the command prompt: “2) Partition data and visualize sequence lengths”. It shows in a figure screen the “Sequence Lengths vs Frequency” generated from the video data. Note: Excessive sequence length will make to take longer to process and less accuracy in the AI model

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

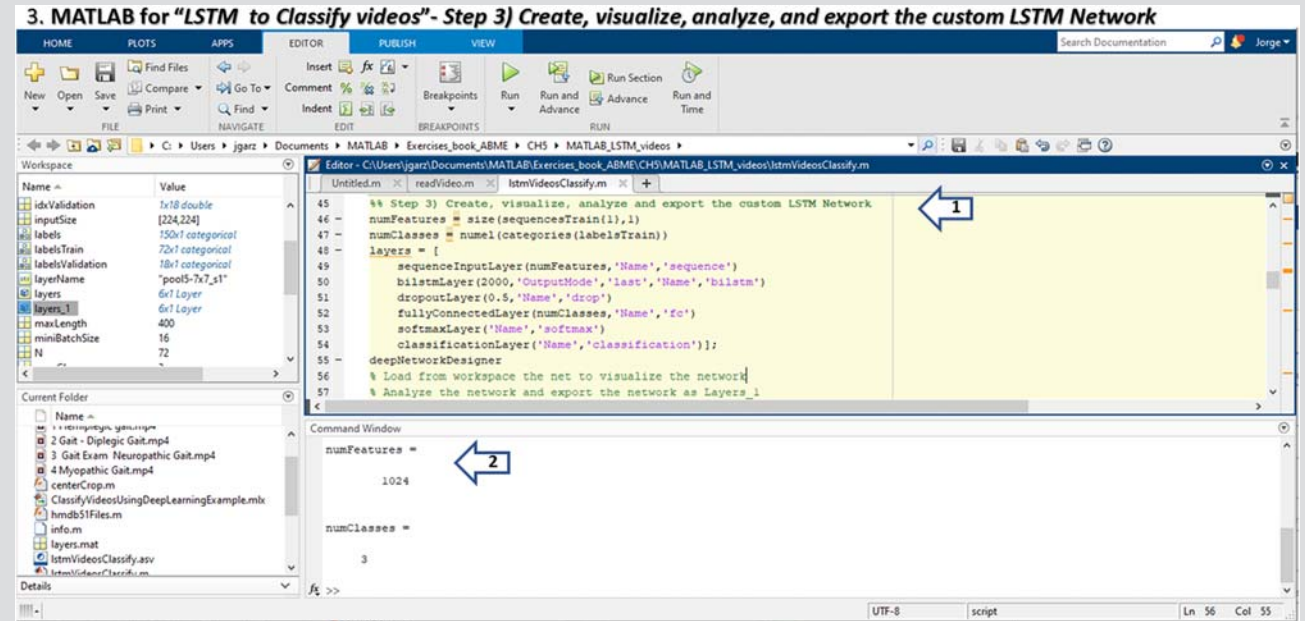
(Continued)

(Continued)

Slide Description

Screen figure

3 MATLAB for “LSTM to Classify videos”—Step 3) Create, visualize, analyze, and export the custom LSTM Network
Copy and paste in the command prompt: “Step 3) Create, visualize, analyze, and export the custom LSTM Network.” Results showed in a command prompt: “numFeatures = 1024” and “numClasses = 3” in the video dataset. Then, call the “Deep Network Analyzer”

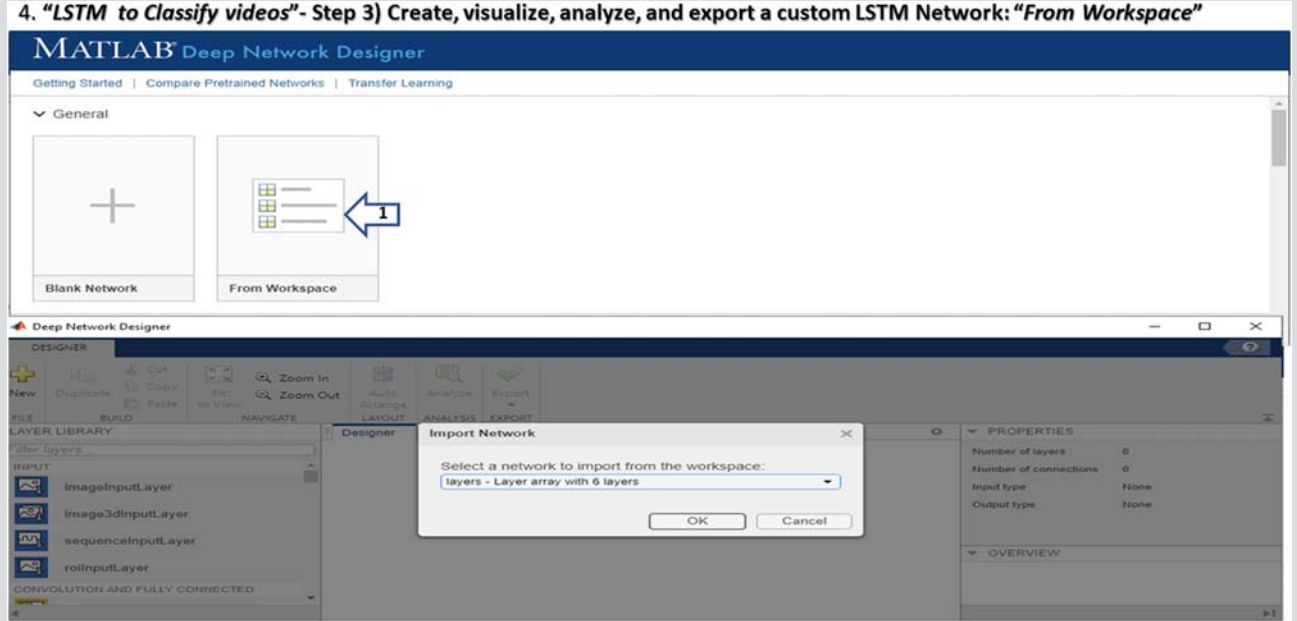


Copy and paste in the command prompt: “Step 3) Create, visualize, analyze, and export the custom LSTM Network”. Results showed in a command prompt: “numFeatures=1024” and “numClasses=3” in the video dataset. Then, call the “Deep Network Analyzer”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

4

MATLAB “LSTM to Classify videos”—Step 3) Create, visualize, analyze, and export a custom LSTM Network: “From Workspace”
In the “Deep Network Designer” initial screen Select “From Workspace,” then in the next screen select “Layers array with 6 layers” and accept by clicking the “Ok” button



In the “Deep Network Designer” initial screen Select “From Workspace”, then in the next screen select “Layers array with 6 layers” and accept clicking in the “Ok” button.

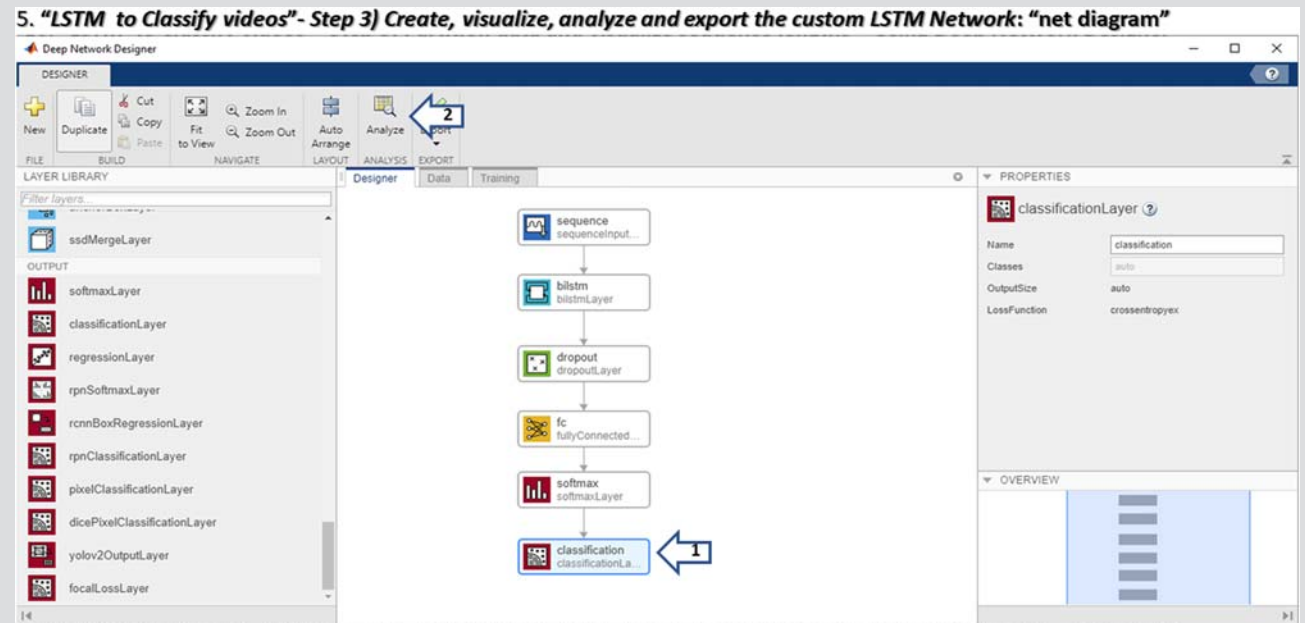
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 5 Description
MATLAB "LSTM to Classify videos"—Step 3) Create, visualize, analyze, and export the custom LSTM Network: "net diagram"
Visualize in the "Deep Network Designer" the "custom LSTM net with 6 layers." Finally, press the "Analyze" button

Screen figure



Visualize in the "Deep Network Designer" the "custom LSTM net with 6 layers". Finally, press the "Analyze" button

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

6


MATLAB “LSTM to Classify videos”—Step 3) Create, visualize, analyze and export the custom LSTM Network: “Net Analyzer”
The “Deep Learning Network Analyzer” must show the “6 layers, with 0 warnings and 0 errors.”
Close and go back to the “MATLAB workspace” screen

6. “LSTM to Classify videos”- Step 3) Create, visualize, analyze and export the custom LSTM Network: “Net Analyzer”

Deep Learning Network Analyzer

Network from Deep Network Designer
Analysis date: 18-Jun-2020 17:13:32

6 layers 2 warnings 3 errors



ANALYSIS RESULT				
Name	Type	Activati...	Learnables	
1 sequence Sequence input with 1 dimensions	Sequence Input	1	-	
2 bilstm BiLSTM with 2000 hidden units	BiLSTM	4000	InputWeights RecurrentWeights	16000*1 16000*2000
3 dropout 50% dropout	Dropout	4000	-	
4 fc 3 fully connected layer	Fully Connected	3	Weights Bias	3*4000 3*1
5 softmax softmax	Softmax	3	-	
6 classification crossentropyex	Classification Output	-	-	

The “Deep Learning Network Analyzer” must show the “6 layers, with 0 warnings and 0 errors”. Close and go back to the “MATLAB workspace” screen.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

7 MATLAB “LSTM to Classify videos”—Step 4) Specify options, train LSTM Network and classify accuracy: “Net Training”
Copy and paste in the command prompt: “Step 4) Specify options, train LSTM Network and classify accuracy.” It will open the “Training Progress” screen. Note: It will take some time for the training based on 3 classes and amount of videos, size of them, “MaxEpochs” specified and the speed of hardware available

7. “LSTM to Classify videos”- Step 4) Specify options, train LSTM Network and classify accuracy: “Net Training”

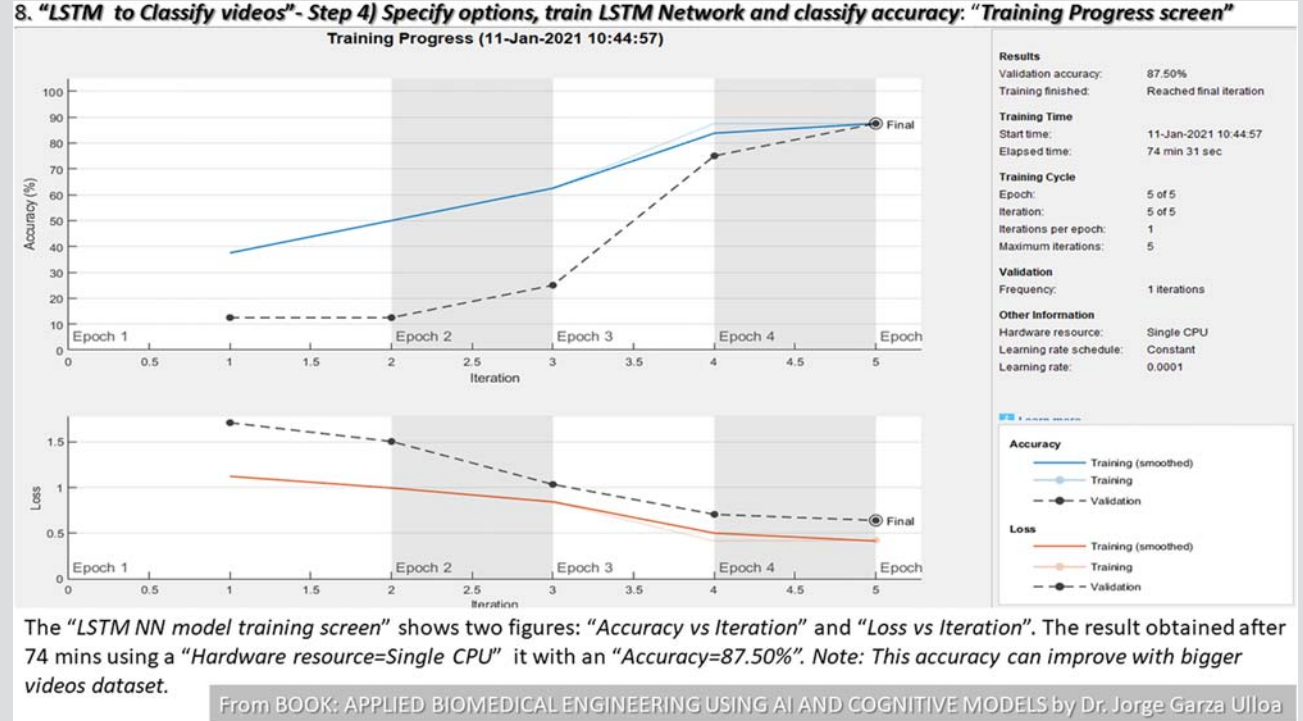
```
60 %% Step 4) Specify options, train LSTM Network and classify accuracy
61 miniBatchSize = 16; numObservations = numel(sequencesTrain);
62 numIterationsPerEpoch = floor(numObservations / miniBatchSize);
63 options = trainingOptions('adam', ...
64     'MiniBatchSize', miniBatchSize, ...
65     'InitialLearnRate', 1e-4, ...
66     'GradientThreshold', 2, ...
67     'Shuffle', 'every-epoch', ...
68     'ValidationData', {sequencesValidation, labelsValidation}, ...
69     'ValidationFrequency', numIterationsPerEpoch, ...
70     'Plots', 'training-progress', ...
71     'MaxEpochs', 5, ...
72     'Verbose', false);
73 [netLSTM, info] = trainNetwork(sequencesTrain, labelsTrain, layers, options);
74 save netLSTM.mat; % save LSTM net
75 %Calculate the classification accuracy of the network on the validation set. Use the same mini-batch size as for the training.
76 YPred = classify(netLSTM, sequencesValidation, 'MiniBatchSize', miniBatchSize);
77 YValidation = labelsValidation; accuracy = mean(YPred == YValidation)
```

Copy and paste in the command prompt: “Step 4) Specify options, train LSTM Network and classify accuracy”. It will open the “Training Progress” screen. Note: It will take some time for the training based on 3 classes and amount of videos, size of them, “MaxEpochs” specified and the speed of hardware available.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

MATLAB “LSTM to Classify videos”—Step 4) Specify options, train LSTM Network, and classify accuracy: “Training Progress screen”

The “LSTM NN model training screen” shows two Figures: “Accuracy versus Iteration” and “Loss versus Iteration.” The result obtained after 84 min using a “Hardware resource = Single CPU” it with an “Accuracy = 87.50%.” Note: This accuracy can improve with bigger videos dataset



(Continued)

(Continued)

Slide 9 Description
"LSTM to Classify videos"—Step 5)
Assembly: Pretrained GoogleNet
net and LSTM for Video
Classification: "Assembly specs"
The "Assembly of both networks:
Pretrained GoogleNet net and a
custom LSTM for Video
Classification is made in this steps
and can be visualized as indicated
in the next 2 slides." A pause is in
this step, switch to the "Deep
Network Designer" please press
space to continue

Screen figure

9. "LSTM to Classify videos"- Step 5) Assembly: Pretrained GoogleNet net and LSTM for Video Classification: "Assembly specs"

```
%% Step 5) Assembly: Pretrained GoogleNet net and LSTM for Video Classification
conLayers = LayerGraph(netCNN);
% In the DeepNetworkDesigner to visualize what layers will be removed
for ind = 1, pause; disp(ind); end; % Pause resume with enter
% Remove the 5 Layers from GoolgeNet one at the beginning and 4 of the end
layerNames = ['fc6' 'pool5' 'conv_5_1' 'conv_5_2' 'conv_5_3'];
conLayers = removeLayers(conLayers, layerNames);
% Add Sequence Input Layer with normalization
inputSize = netCNN.Layers(1).inputSize(1:2);
averageImage = netCNN.Layers(1).mean;
inputLayer = sequenceInputLayer(inputSize, ...
    'normalization', 'resnetnet', ...
    'mean', averageImage, ...
    'name', 'input');
% Add the sequence input layer and sequenceFoldingLayers at the beginning at
layers = [inputLayer sequenceFoldingLayer('name', 'fold')];
% Connect the layers
igraph = addLayers(conLayers, layers);
% Connect layers (igraph, 'fold/out', 'conv5_1'); % Connect layers
inputLayers = netLSTM.Layers(inputLayers); % {}
% Add 4 layers at the end and connect them
layers = [
    sequenceUnfoldingLayer('name', 'unfold')
    flattenLayer('name', 'flatten')
    lstmLayer];
igraph = addLayers(igraph, layers);
% Connect the layers
igraph = connectLayers(igraph, 'fold/miniBatchSize', 'unfold/miniBatchSize');
igraph = connectLayers(igraph, 'pool5/mat_0', 'net_lstm');
% Assemble network for prediction using the assembleNetwork function.
net = assembleNetwork(igraph); save net.mat;
% In the DeepNetworkDesigner visualize and analyze the ensemble network
```

Command Window

```
igraph = addLayers(igraph, layers);
% Connect the layers
igraph = connectLayers(igraph, 'fold/miniBatchSize', 'unfold/miniBatchSize');
igraph = connectLayers(igraph, 'pool5/mat_0', 'net_lstm');
% Assemble network for prediction using the assembleNetwork function.
net = assembleNetwork(igraph); save net.mat;
% In the DeepNetworkDesigner visualize and analyze the ensemble network
```

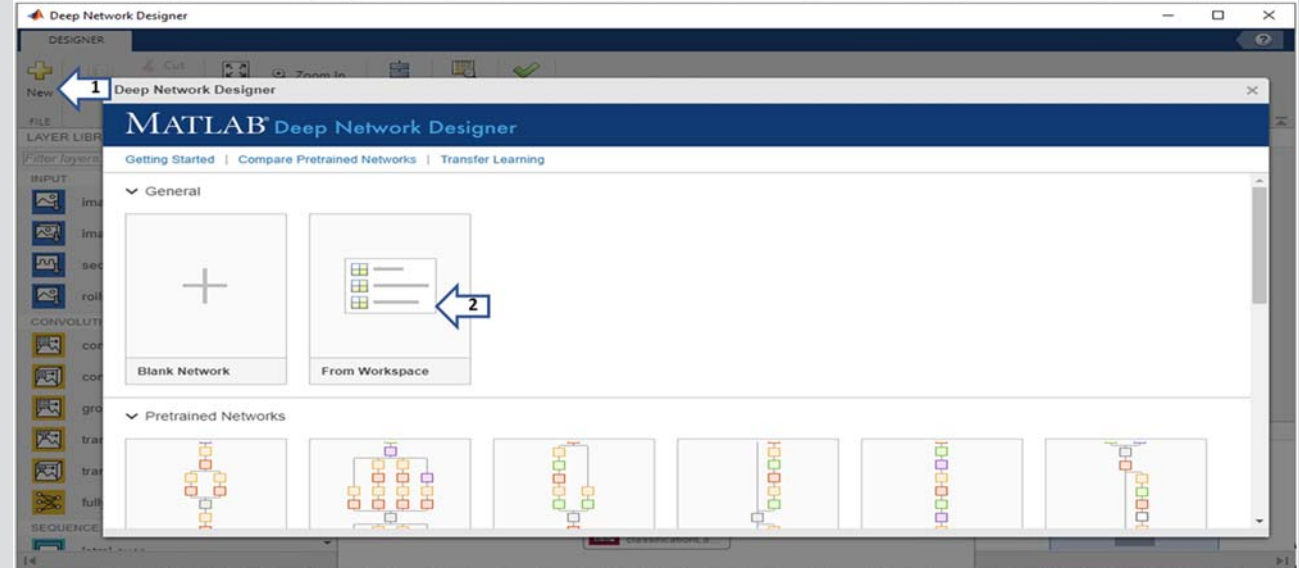
The "Assembly of both networks: Pretrained GoogleNet net and a custom LSTM for Video Classification is made in this steps and ca be visualized as indicated in the next 2 slides". Note: A pause is in this step switch to the "Deep Network Designer" please press space to continue.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

10

MATLAB “LSTM to Classify videos”—Step 5) Assemble: Pretrained GoogleNet net and LSTM for Video Classification: “Original GoogLeNet”
In the “Deep Network Designer” that shows the custom LSTM NN select “New,” then in the next screen select “From Workspace” as indicated in the slide

10. “LSTM Classify videos”- Step 5) Assembly: Pretrained GoogleNet net & LSTM for Video Classification: “Original GoogLeNet”



In the “Deep Network Designer” that show the custom LSTM NN select “New”, then in the next screen select “From Workspace” as indicated in the slide.

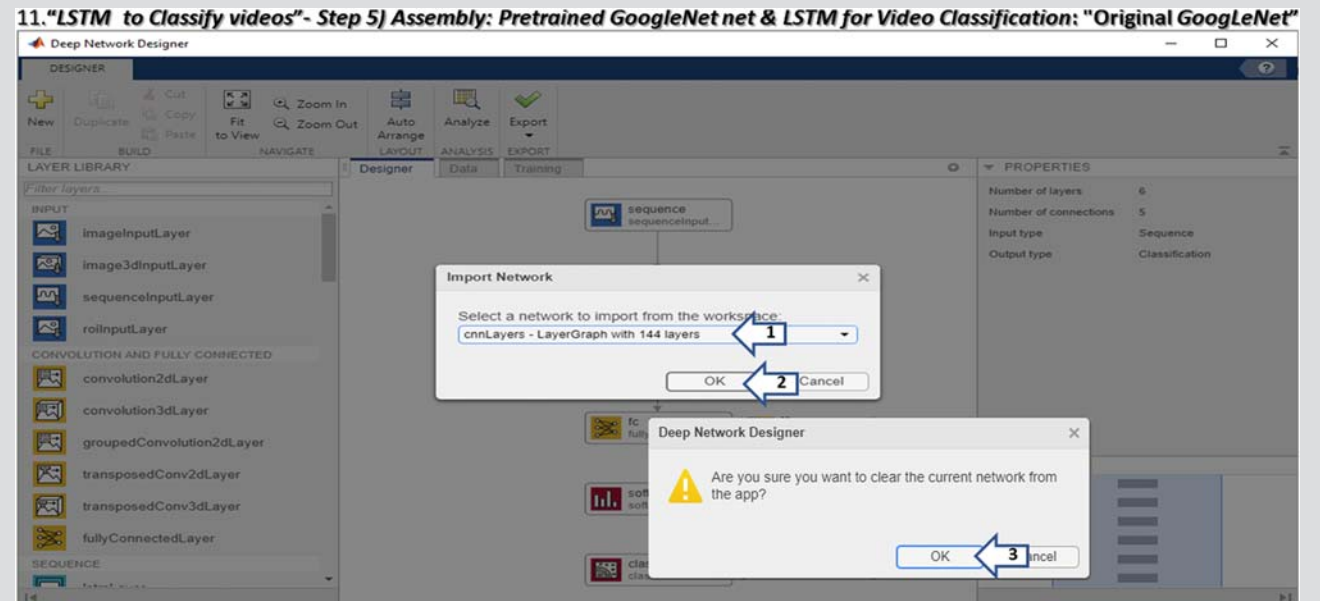
From BOOK: APPLIED BIOMEDICAL ENGINEERING: USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 11 Description
MATLAB “LSTM to Classify videos”—Step 5) Assemble: Pretrained GoogleNet net and LSTM for Video Classification: “Original GoogLeNet”
In the “Deep Network Designer” in the “Import Network” dialog select the assembly net “cnnLayers—LayerGraph with 144 Layers,” then click on the “OK” button. Finally, “accept to clear the current network from the app?”

Screen figure

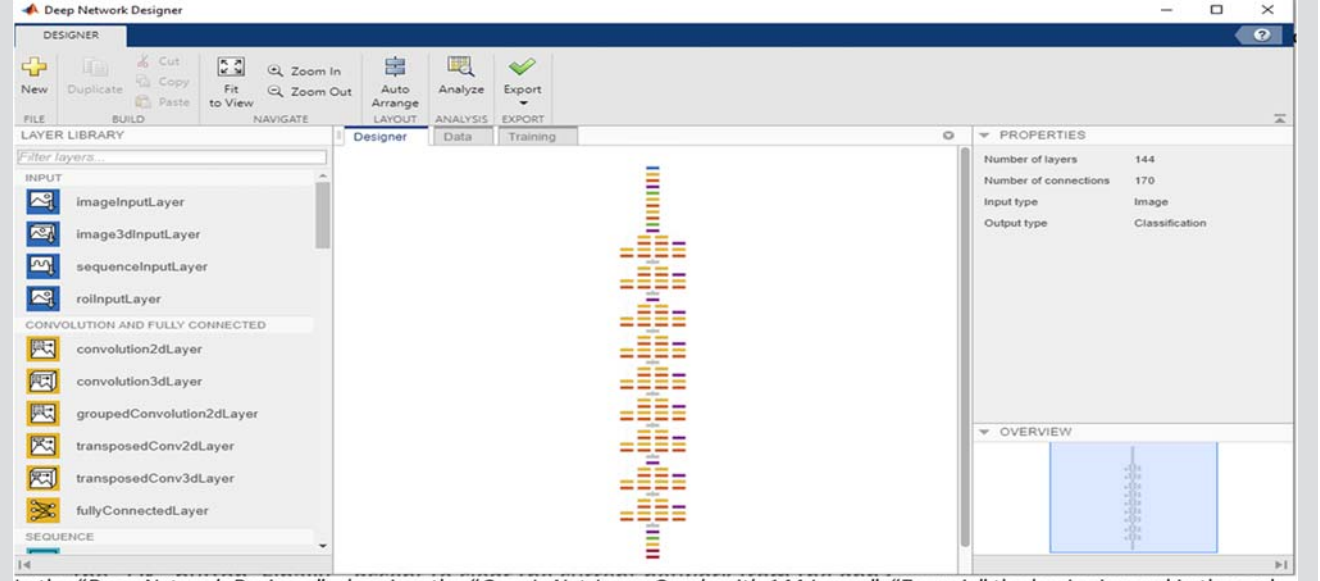


In the “Deep Network Designer” in the “Import Network” dialog select the assembly net “cnnLayers - LayerGraph with 144 Layers”, then click on the “OK” button. Finally, “accept to clear the current network from the app?”

12

“LSTM to Classify videos”—Step 5) Assembly: Pretrained GoogleNet net and LSTM for Video Classification: “Original GoogLeNet”
In the “Deep Network Designer” showing the “GoogLeNet Layer Graph with 144 Layers,” “Zoom in” the beginning and in the end as shown in the next slide

12. “LSTM to Classify videos”- Step 5) Assembly: Pretrained GoogleNet net & LSTM for Video Classification: “Original GoogLeNet”



In the “Deep Network Designer” showing the “GoogLeNet Layer Graph with 144 Layers,” “Zoom in” the beginning and in the end as shown in the next slide

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

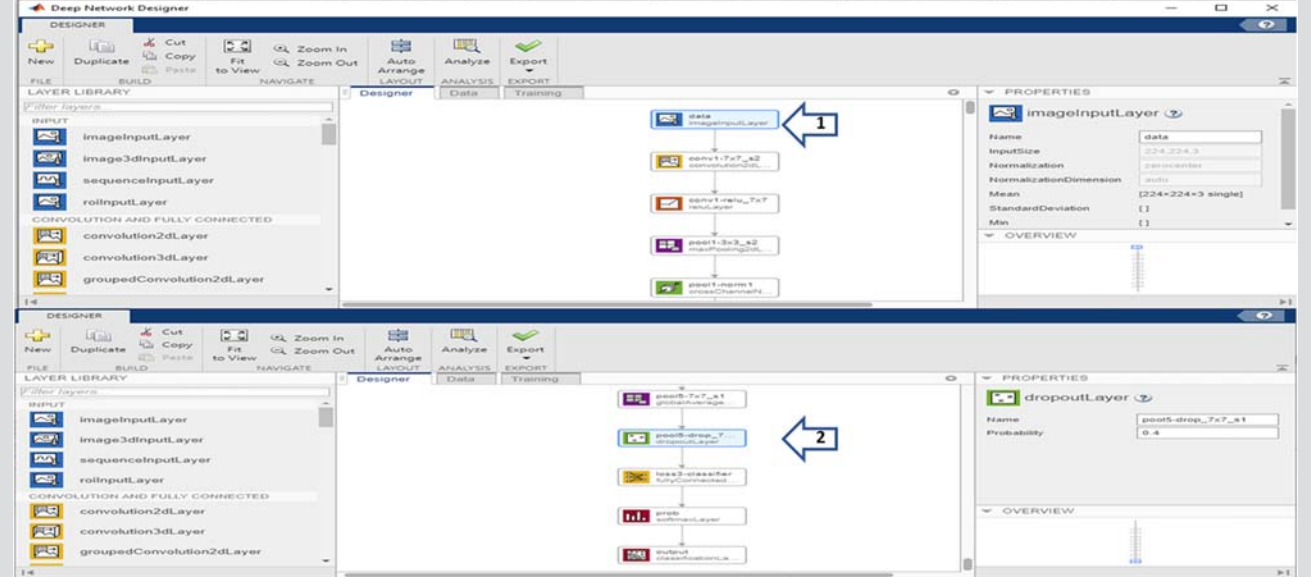
(Continued)

(Continued)

Slide 13 Description
"LSTM to Classify videos" – Step 5) Assembly: Pretrained GoogleNet net and LSTM for Video Classification "Original GoogLeNet"
In the "Deep Network Designer" shows at the top, the "first layer = data (imageInputLayer)" of "GoogLeNet" will be removed, and at the bottom the screen shows the end where the last 4 layers: "pool5-drop_7x7_s1", "loss3-classifier", "prob" and "output" will also be removed and then replaced. Go back to MATLAB and press enter to continue the script

Screen figure

13. "LSTM to Classify videos" - Step 5) Assembly: Pretrained GoogleNet net & LSTM for Video Classification "Original GoogLeNet"

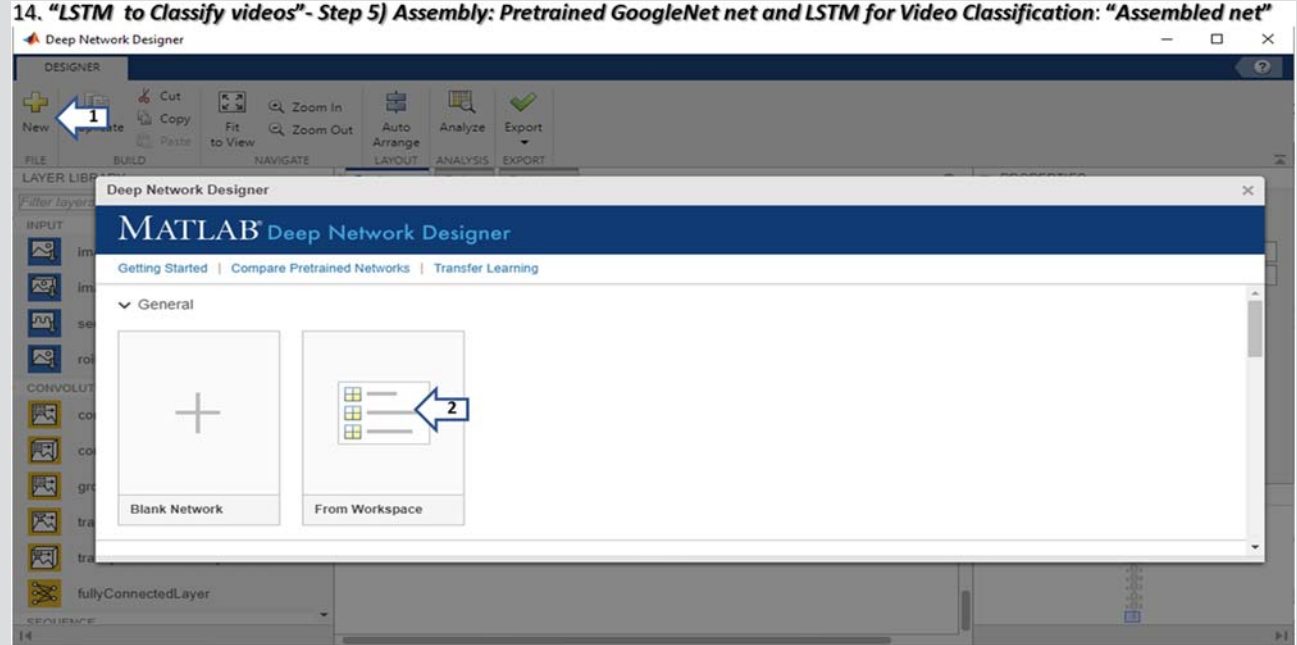


In the "Deep Network Designer" shows at the top, the "first layer = data (imageInputLayer)" of "GoogLeNet" will be removed, and at the bottom screen shows the end where the last 4 layers: "pool5-drop_7x7_s1", "loss3-classifier", "prob" and "output" will also be removed and then replaced. Go back to MATLAB and press enter to continue the script

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

14

“LSTM to Classify videos”—Step 5) Assembly: Pretrained GoogleNet net and LSTM for Video Classification: “Assembled net”
In the “Deep Network Designer” that shows the original “GoogLeNet” select “New,” then in the next screen select “From Workspace” as indicated



In the “Deep Network Designer” that show the original “GoogLeNet” select “New”, then in the next screen select “From Workspace” as indicated.

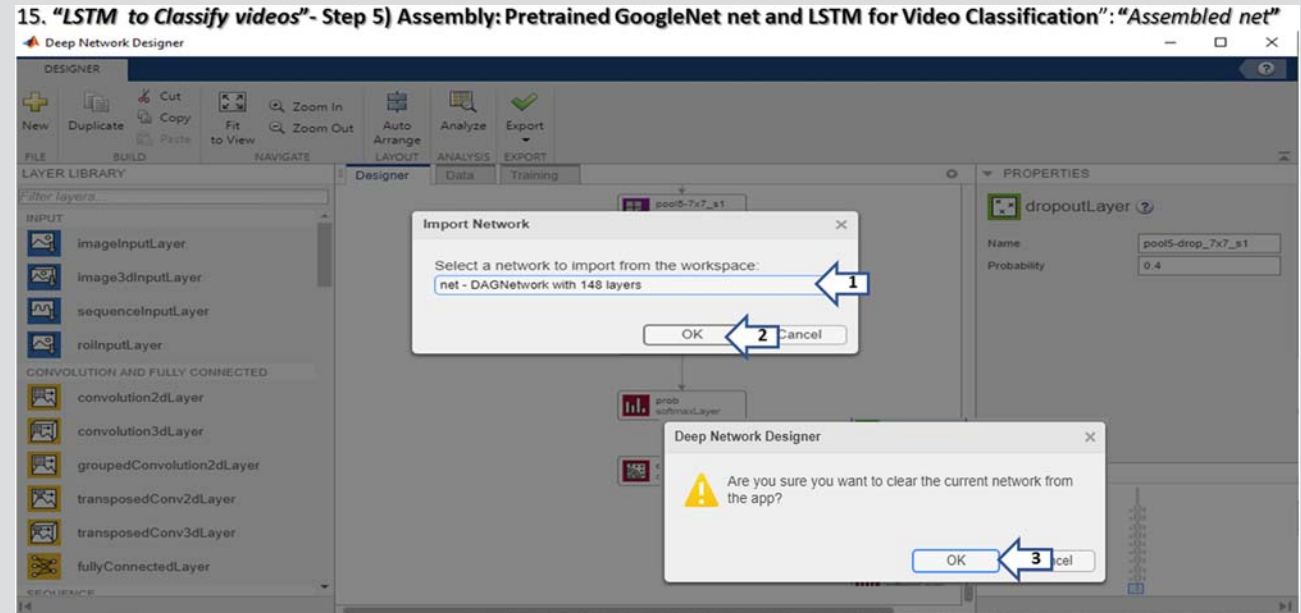
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 15 Description
"LSTM to Classify videos"—Step 5) Assembly: Pretrained GoogleNet net and LSTM for Video Classification: "Assembled net"
In the "Deep Network Designer" in the "Import Network" dialog select the assembled net "net-DAGNetwork with 148 layers," then click on the "OK" button. Finally, "accept to clear the current network from the app?"

Screen figure

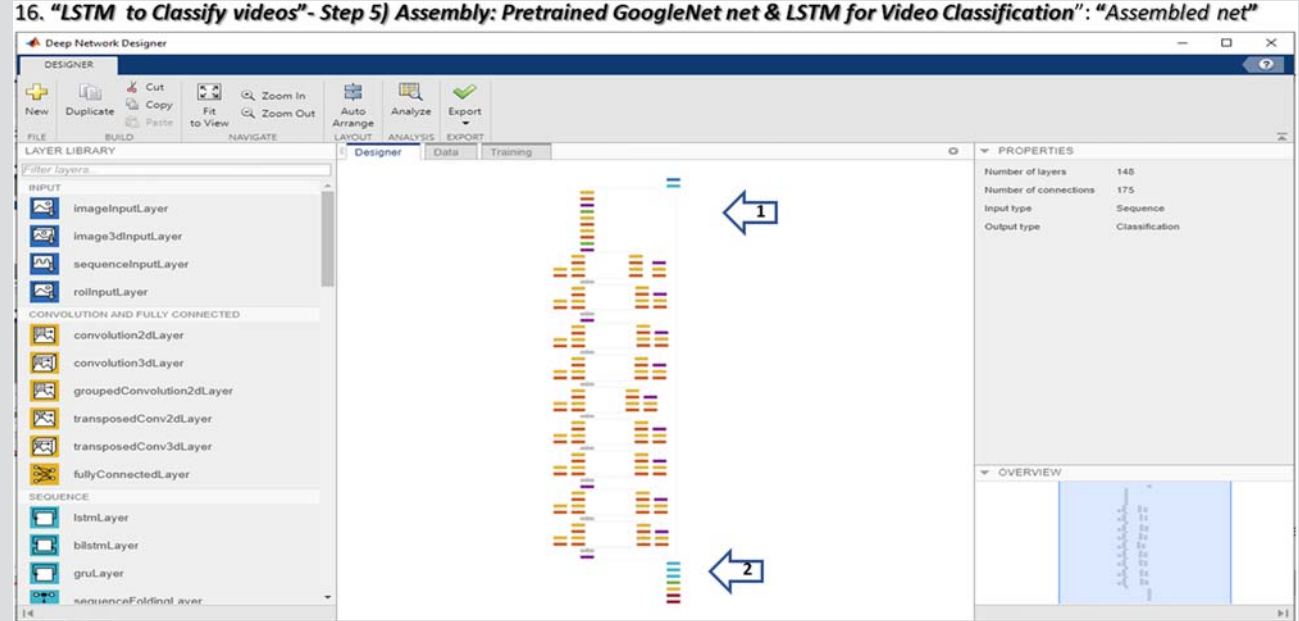


In the "Deep Network Designer" in the "Import Network" dialog select the assembled net "net-DAGNetwork with 148 layers", then click on the "OK" button. Finally, "accept to clear the current network from the app?"

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

16

“LSTM to Classify videos”—Step 5) Assembly: Pretrained GoogleNet net and LSTM for Video Classification: “Assembled net”
In the “Deep Network Designer” showing the “Assembled Net Layer Graph with 148 Layers,” “Zoom in” the beginning and in the end as shown in the next two slides



In the “Deep Network Designer” showing the “Assembled Net Layer Graph with 148 Layers”, “Zoom in” the beginning and in the end as shown in the next two slides

From BOOK: APPLIED BIOMEDICAL ENGINEERING: USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide	Description	Screen figure
17	<p><i>“LSTM to Classify videos”—Step 5) Assembly: Pretrained GoogLeNet net and LSTM for Video Classification: “Beginning of assembled net”</i></p> <p>In the <i>“Deep Network Designer”</i> showing at the top of the ensembled net, the <i>“first layer = input (sequenceInputLayer)”</i> and the <i>“Second layer = fold (sequenceFoldingLayer)”</i> that replace the first layer of the original trained <i>“GoogLeNet”</i></p>	<p>17. “LSTM to Classify videos”- Step 5) Assemble: Pretrained GoogLeNet net and LSTM for Video Classification: “Beginning of net”</p> <p>In the <i>“Deep Network Designer”</i> shows at the top of the ensembled net, the <i>“first layer=input (sequenceInputLayer)”</i> and the <i>“Second layer=fold(sequenceFoldingLayer)”</i> that replace the first layer of the original trained <i>“GoogLeNet”</i></p> <p>From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

“LSTM to Classify videos”—Step 5) Assembly: Pretrained GoogleNet net and LSTM for Video Classification: “The End of net”
 In the “Deep Network Designer” showing the assembled net at the bottom, the “*unfold (sequenceUnfoldingLayer), “Flatten (flattenLayer),” and the “custom LSMT net”* designed and trained in the first part of this research tutorial. Finally press the “Analyze” button

18. “LSTM to Classify videos”- Step 5) Assembly: Pretrained GoogleNet net and LSTM for Video Classification”: “The End of net”

In the “Deep Network Designer” shows the assembled net at the bottom, the “*unfold (sequenceUnfoldingLayer), “Flatten (flattenLayer),” and the “custom LSMT net”* designed and trained in the first part of this research tutorial . Finally press the “Analyze” button

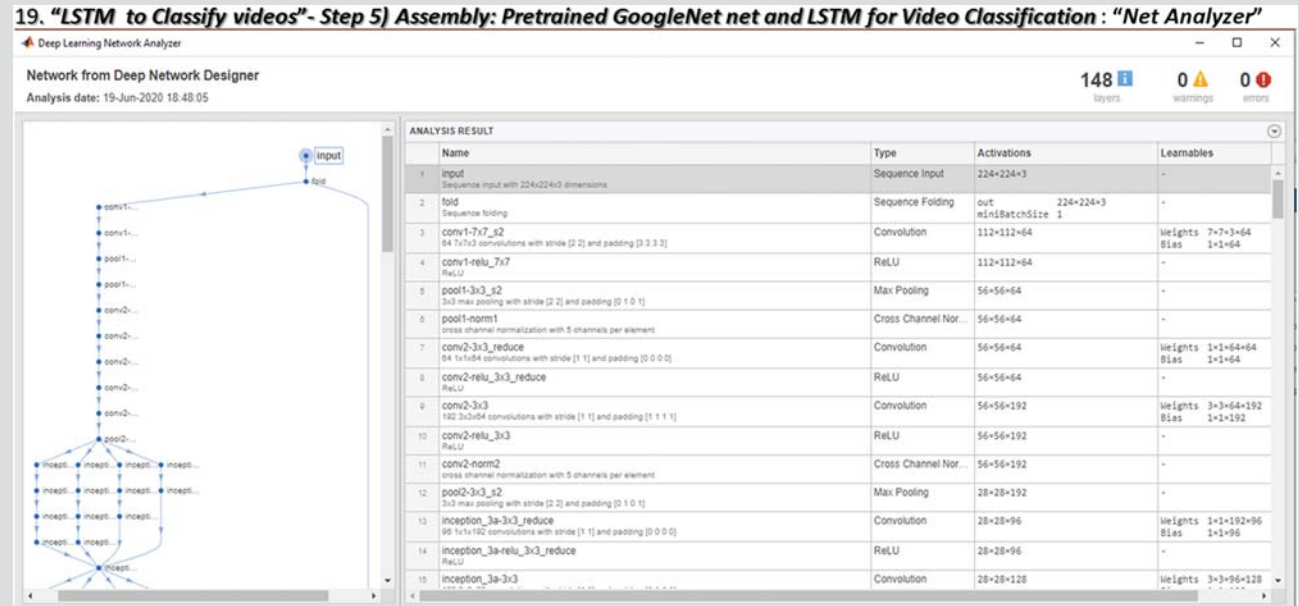
From BOOK: APPLIED BIOMEDICAL ENGINEERING: USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 19 Description
"LSTM to Classify videos"—Step 5)
Assembly: Pretrained GoogleNet
net and LSTM for Video
Classification: "Net Analyzer"
The "Deep Learning Network
Analyzer" must show the result for
the ensembled net with "148 layers,
with 0 warnings and 0 errors."
Close and go back to the screen of
MATLAB workspace, "press enter"
on the keyboard to continue from the
pause

Screen figure



The "Deep Learning Network Analyzer" must shows the result for the ensembled net with "148 layers, with 0 warnings and 0 errors". Close and go back to the screen of MATLAB workspace, "press the enter" in the keyboard to continue from the pause.

20

“LSTM to Classify videos”—Step 6) Classify Using New Video Data to “detect human body movements as falls”

Copy and paste in the command prompt: *“Step 6) Classify Using New Video Data to detect human body movements like falls.”* In this step the video named *“Fall_14.MP4”* is classified for the LSTM NN and it is categorized as *“Fall.”* You can try other videos included in this subdirectory. Close all and exit MATLAB

20. “LSTM to Classify videos”- Step 6) Classify Using New video Data to “detect human body movements as falls”

The screenshot shows the MATLAB environment with the following components:

- Workspace:** A table of variables and their values:

Name	Value
accuracy	0.6250
ans	10x1 cell
averageImage	224x224x3 single
cnnLayers	1x1 LayerGraph
dataFolder	"humanMov"
filename	"Fall_14.mp4"
idx	28x1 string
idx	28
idx	1x28 logical
idxTrain	1x28 double
idxValidation	[24,11,27,26,25,9,4,32]
ind	1
- Editor:** MATLAB code for Step 6:

```
%% Step 6) Classify Using New video Data
% Read and center-crop the video "pushup.mp4" using the same steps as before.
filename = "Fall_14.mp4";
video = readVideo(filename);
% Classify the video using the assembled network. The classify function expects a cell array containing the input video.
video = centerCrop(video,inputSize);
YPred = classify(net,(video));
```
- Command Window:** Execution output:

```
>> % Read and center-crop the video "pushup.mp4" using the same steps as before.
filename = "Fall_14.mp4";
video = readVideo(filename);
% Classify the video using the assembled network. The classify function expects a cell array containin
video = centerCrop(video,inputSize);
YPred = classify(net,(video));

YPred =
    categorical
         fall
```
- Current Folder:** A directory listing showing files like 'humanMov', 'Fall_13.MP4', and 'Fall_14.MP4'.
- Video Preview:** A small window showing a person falling.

Copy and paste in the command prompt: *“Step 6) Classify Using New video Data to detect human body movements like falls”*. In this step the video named *“Fall_14.MP4”* is classified for the LSTM NN and it is categorized as *“Fall”*. You can try other videos included. Close all and exit MATLAB.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

Using “Obtain an LSTM model from MATLAB Deep Learning Toolbox using a Deep Network Designer to create an LSTM NN to Classify videos about human body movements and detect walking human falls,” new videos can be classified using the AI model obtained to assign their categories or classes.

Recommendation

This kind of “LSTM model” can be applied to a diversity of video datasets to be analyzed in different topics in biomedical engineering.

There are basically four types of connection in “RCNN”; these are “bottom-up (B),” “bottom-up lateral (BL),” “bottom-up top-down (BT)” and “bottom-up lateral top-down (BLT)” as shown at Fig. 6.5B” Combining these four types of connections between layers allow “RCNN” to have the “ability to recognize objects using computer vision” in different applications suggesting the “recurrent connections in biological brains” [14].

6.2.4 Recurrent convolutional neural networks

“Recurrent convolutional neural networks (RCNN)” are basically a “convolutional neural network (CNN)” and a “recurrent neural network (RNN),” as indicated in Fig. 6.5A. Where the “RNN” includes lateral and feedback connections. “Feed forward neural networks” provide the dominant model of how the brain performs visual object recognition. However, these networks lack the lateral and feedback connections, and the resulting “recurrent neuronal dynamics,” of the ventral visual pathway in the human and nonhuman primate brain [13].

6.2.5 Regional-Convolutional Neural Network Object detection in AI models

“Computer vision” is an interdisciplinary field of “Computer Science” and “Electrical and Computer Engineering” that has grown exponentially thanks to the application of “Deep Learning” in a wide diversity of applications. We focus in this section on “object detection and their application in Biomedical Engineering.” The difference between “classification and object detection” is a “bounding box or many of them to indicate a “Region” for objects of interest within the images,” that we know before training. The standard “CNN” cannot handle different spatial locations with different ratios, because this will require a huge number of regions taking an exceptionally long computational process requiring

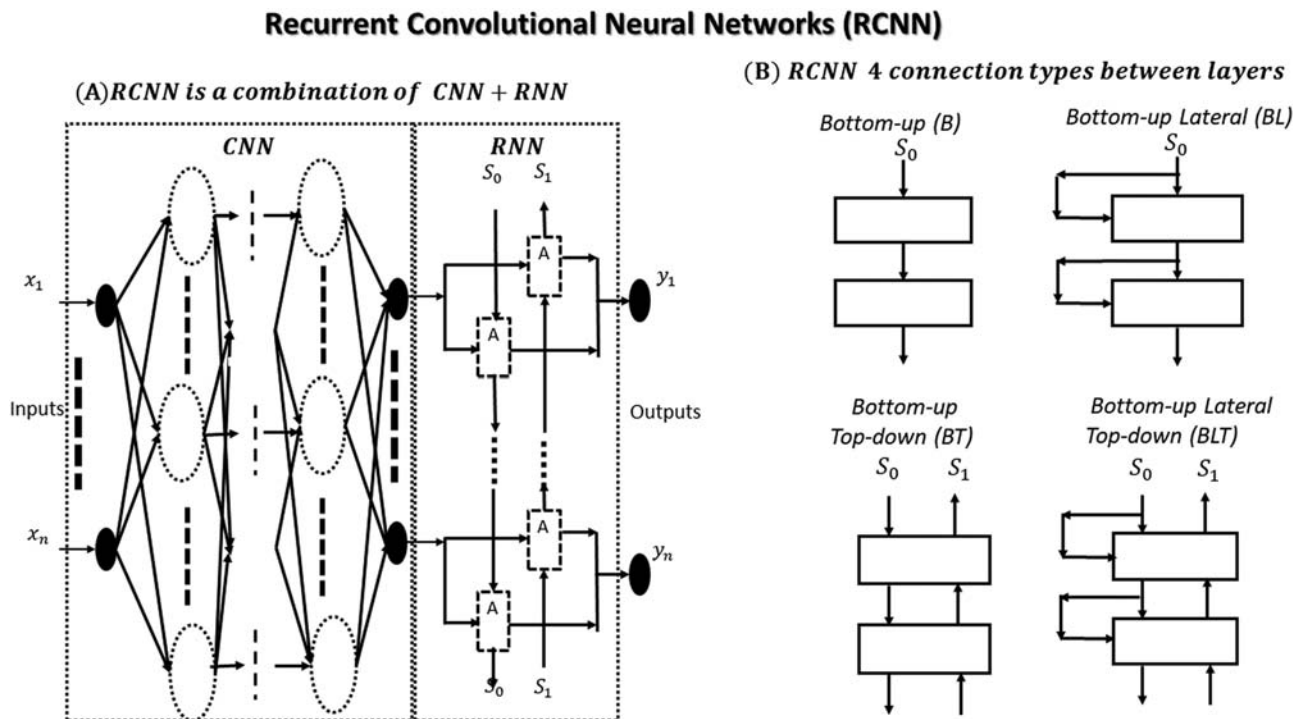


FIGURE 6.5 Recurrent Convolutional Neural Networks (RCNN): (A) RCNN is a combination of CNN + RNN and (B) RCNN basically has four types of connections between layers.

enormously powerful hardware. There are some “*AI procedures to resolve this problem*” such as “*Regional-Convolutional Neural Network (R-CNN)*,” “*Fast R-CNN*,” “*YOLO*,” and others.

- “*Regional-Convolutional Neural Network (R-CNN)*” is an algorithm that uses a selective search to extract only “*n*” regions that avoid classifying a huge number of regions into their respective classes. The main drawback is a fixed algorithm, that cannot be implemented in real-time applications.
- “*Fast Regional-Convolutional Neural Network (Fast R-CNN)*” is like the “*R-CNN*” but the “*convolution is made only one time per image and a feature map is generated from it.*” “*Fast R-CNN*” identifies the region of proposals and warps them into squares and using a combination of “*Pooling layer*” reshapes them into a fixed size so that they can be fed into a fully connected layer. Then, a “*Softmax layer*” and a “*Classification Layer*” is applied to predict the class of the proposed region and the offset values for the bounding box, as shown in the “*Research 6.2, section 6.2.5.1 MATLAB Deep Learning Toolbox to create an R-CNN for object detection of breast tumor in mammogram.*”
- “*You Only Look Once (YOLO)*” is a real-time object detection that is based on a single neural network to the full image. This network divides the image into regions and predicts bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities.

Some currently used examples of “*Regional-Convolutional Neural Network (R-CNN)*” in “*Biomedical Engineering*” are:

- “*Deep learning framework based on integration of S-Mask R-CNN and Inception-v3 for ultrasound image-aided diagnosis of prostate cancer*” by Liu et al. [72]. The ultrasound images of the prostate sometimes come with serious speckle noise, low signal-to-noise ratio, and poor detection accuracy. To overcome this shortcoming, a deep learning model that integrates S-Mask R-CNN and Inception-v3 in the ultrasound image-aided diagnosis of prostate cancer is proposed in this paper. The improved S-Mask R-CNN was used to realize the accurate segmentation of prostate ultrasound images and generate candidate regions.
- “*FibeR-CNN: Expanding mask R-CNN to improve image-based fiber analysis*” by Frei and Kruijs [73]. Fiber-shaped materials (e.g., carbon nanotubes) are of great relevance, due to their unique properties but also the health risk they can impose. They propose the use of region-based convolutional neural networks “*R-CNNs*” to automate this task. “*Mask R-CNN*,” most

widely used for semantic segmentation tasks, is prone to errors when it comes to the analysis of fiber-shaped objects. Hence, a new architecture “*FibeR-CNN*” is introduced and validated. FibeR-CNN combines two established R-CNN architectures (Mask and Keypoint R-CNN) and adds additional network heads for the prediction of fiber widths and lengths. As a result, FibeR-CNN can surpass the mean average precision of Mask R-CNN by 33% (11 percentage points) on a novel test data set of fiber images.

- “*Computer-aided detection of COVID-19 from X-ray images using multi-CNN and Bayesnet classifier*” by Abraham and Nair [15]. The gold standard for diagnosing “*COVID-19*” is a “*reverse transcription-polymerase chain reaction (RT-PCR)*” test. However, the facility for “*RT-PCR test*” is limited, which causes early diagnosis of the disease difficult. Easily available modalities like X-rays can be used to detect specific symptoms associated with “*COVID-19*.” Pretrained convolutional neural networks are widely used for computer-aided detection of diseases from smaller datasets. This paper investigates the effectiveness of “*multi-CNN*,” a combination of several “*pre-trained CNNs*,” for the automated detection of “*COVID-19*” from X-ray images.

The standard “*CNN*” cannot handle different spatial locations with different ratios, because this will require a huge number of regions taking an exceptionally long computational process requiring enormously powerful hardware. By contrast, “*R-CNN*” allows object detection, and it is very useful in biomedical engineering images analysis.

6.2.5.1 Research 6.2 Regional-CNN model for object detection of breast tumor in mammogram

6.2.5.1.1 Case for research

“*Obtain an R-CNN model from MATLAB Deep Learning Toolbox classification for object detection of breast tumor in mammogram.*”

6.2.5.1.2 General objective

Apply “*MATLAB Deep Learning Toolbox*” “to define, build, and train an “*R-CNN Neural Network model to classify for object detection of breast tumor in mammograms.*”

6.2.5.1.3 Specific objectives

1. Open MATLAB for “*R-CNN for object detection as breast tumor*”—Load the table of labels of mammogram images, and MATLAB path to this directory.

2. Use the “*deepNetworkDesigner*” to visualize the layer of the “*custom R-CNN*,” see the properties for each layer, and “*analyze the R-CNN net*.”
3. Define Options for the Training and “*Train R-CNN for object detection of breast tumors*.”
4. Testing new “*mammogram images*” with the “*R-CNN model to detect tumor breast*.”
5. Display the “*mammogram image with its breast tumor region*” applying “*R-CNN object detection with its best probability or score*.”

6.2.5.1.4 Background for “breast tumors in mammogram”

The image of the breast is taken by “*low-dose X-rays*” bioinstruments, known as a “*mammogram*.” The images are taken on a black background and the breast is shown in grays and whites; denser tissues, connective tissues, and glands show up in white. Any area that does not look like a normal tissue show as areas of white as high-density tissue, and it is important to take note of their size, shape, and edges. A “*lump*” or “*tumor*” shows up as a white area on a “*mammogram*.” Breast abnormalities* are usually a “*mass*” detected in a breast, which could be

a “*tumor*,” “*cyst*,” “*calcifications*,” “*fibroadenomas*,” or “*scar tissue*.”

6.2.5.1.5 Dataset

The dataset consists of two folders with 10 low-dose X-rays of patients with breast tumors as shown at figure Fig. 6.6: the top folder under the name “*breastTumor*” contains “*10 original breast X-rays*”, and the bottom folder under the name “*breastTumorLabel*” has “*the same 10 breast X-rays labeled, indicating inside a rectangle the region of breast to be trained for object detection*”.

Note: This images dataset is available in the companion directory of the book, in the following folder “*...Exercises_book_ABME\CH6\MATLAB_R-CNN*.”

6.2.5.1.6 Procedure

The steps to obtain an “*R-CNN model from MATLAB Deep Learning Toolbox classification for object detection of breast tumor in mammogram*” [16] are summarized in Table of slides 6.2, and each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

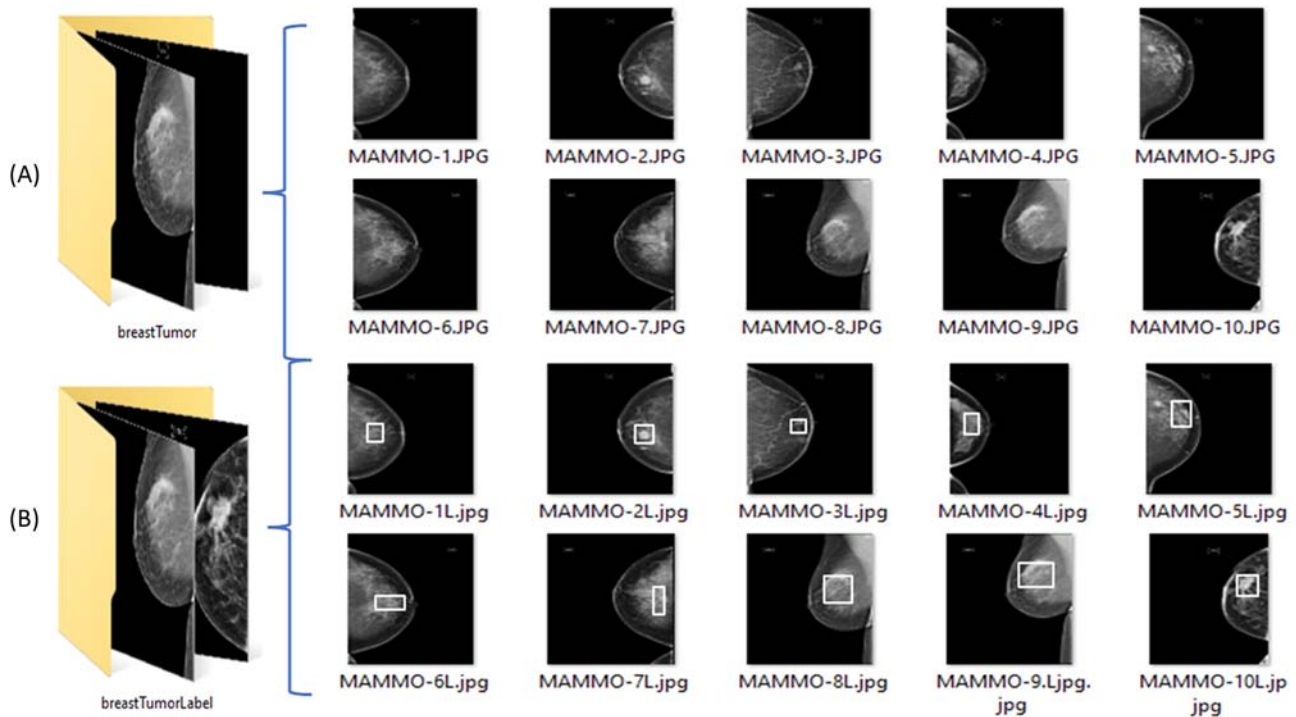


FIGURE 6.6 The image dataset for breast tumor is organized in two folders: (A) “*breastTumor*” containing “*10 original breast X-rays*” and (B) “*breastTumorLabel*” with the same 10 breast X-rays labeled, indicating inside “*a rectangle the region of breast to be trained for object detection*.”

Table of slides 6.2 Steps for MATLAB Deep Learning Toolbox to R-CNN model from MATLAB Deep Learning Toolbox classification for object detection of breast tumor in mammogram.

Slide Description

Screen figure

1 Open MATLAB for “R-CNN for object detection as breast tumor”—Step 1) Load table of labels, R-CNN layers. and add images path
Go to the directory “... \Exercises_book_ABME\CH6\MATLAB_R-CNN.” Open the script “detectBreastTumorRCNN.m.” Copy and paste in the command prompt: “Step 1) Load table of labels, R-CNN layers and add Mammogram path.” Double click the variables: “breastTumor 10x2 of imageFilename and breastTumorXYWH” as positions from left bottom “corner X = horizontal & Y = Vertical, W = width, and H = height” and “layersRCNN.” Finally, call “deepNetworkDesigner” to visualize the net and analyze it

1. Open MATLAB for “R-CNN for object detection breast tumor”- Step 1) Load table of labels, R-CNN layers and add image path

The screenshot shows the MATLAB environment with the following components:

- Workspace:** Contains variables `breastTumor` (10x2 table), `imDir` (string), and `layersRCNN` (15x1 Layer).
- Editor:** Shows the script `detectBreastTumorRCNN.m` with the following code:


```

%% Step 1) Load table of labels, R-CNN layers and add Mammogram path
load breastTumor; % Load Labels Table
load layersRCNN; % Load RCNN layers
imDir=append(pwd, '\breastTumor'); % imDir is images breastTumor
addpath(imDir); % Add Breast tumor folder to the path
deepNetworkDesigner % to visualize and analyze the net
      
```
- Command Window:** Shows the execution of the script:


```

>> %% Step 1) Load table of labels, R-CNN layers and add Mammogram path
load breastTumor; % Load Labels Table
load layersRCNN; % Load RCNN layers
imDir=append(pwd, '\breastTumor'); % imDir is images breastTumor
addpath(imDir); % Add Breast tumor folder to the path
deepNetworkDesigner % to visualize and analyze the net
      
```
- Current Folder:** Shows the directory structure including `breastTumor`, `breastTumorLabel`, `other`, `breastTumor.mat`, `detectBreastTumorRCNN.asv`, `detectBreastTumorRCNN.m`, `layersRCNN.mat`, and `MAMMO-test.JPG`.
- LayersRCNN:** A list of 15 layers:
 - 1x1 ImageInputLayer
 - 1x1 Convolution2DLayer
 - 1x1 MaxPooling2DLayer
 - 1x1 ReLU Layer
 - 1x1 Convolution2DLayer
 - 1x1 ReLU Layer
 - 1x1 AveragePooling2DLayer
 - 1x1 Convolution2DLayer
 - 1x1 ReLU Layer
 - 1x1 AveragePooling2DLayer
 - 1x1 FullyConnectedLayer
 - 1x1 ReLU Layer
 - 1x1 FullyConnectedLayer
 - 1x1 SoftmaxLayer
 - 1x1 ClassificationOutputLayer

Go to the directory “... \Exercises_book_ABME\CH6\MATLAB_R-CNN”. Open the script “detectBreastTumorRCNN.m”. Copy and paste in the command prompt: “Step 1) Load table of labels, R-CNN layers and add Mammogram path”. Double click the variables: “breastTumor 10x2 of imageFilename and breastTumorXYWH” as positions from left bottom “corner X = horizontal & Y = Vertical, W = width, and H = height” and “layersRCNN”. Finally, call “deepNetworkDesigner” to visualize the net and analyze it.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

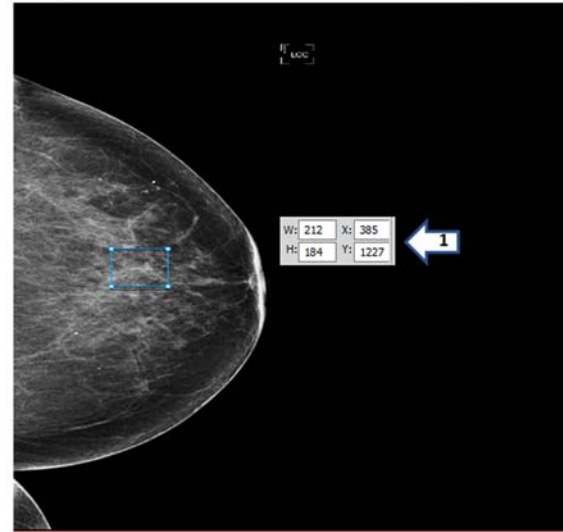
Slide 2 Description

MATLAB for “R-CNN for object detection as breast tumor”—
Notes: About the breast tumor labels variable.
Notes: The MATLAB variable “breastTumor” is a table 10 × 2 that has the “imageFilename” and position of the breast tumor of 2117 × 2606, and the object to detect is the “tumor breast formation” indicated with a rectangle as “XYWH” for each of the 10 images provided in the folder “breastTumorLabel.” The table can be generated by MATLAB tool in “Computer Vision Toolbox” known as “Image Labeler”

Screen figure

2. MATLAB for “R-CNN for object detection as breast tumor”- Notes: About the breast tumor labels variable

a) Image: “MAMMO-1L.JPG resolution: W=2117, H=2605”



b) Variable “breastTumor is a table 10 x 2”

The image shows the MATLAB variable viewer for the variable 'breastTumor'. It is a 10x2 table with the following data:

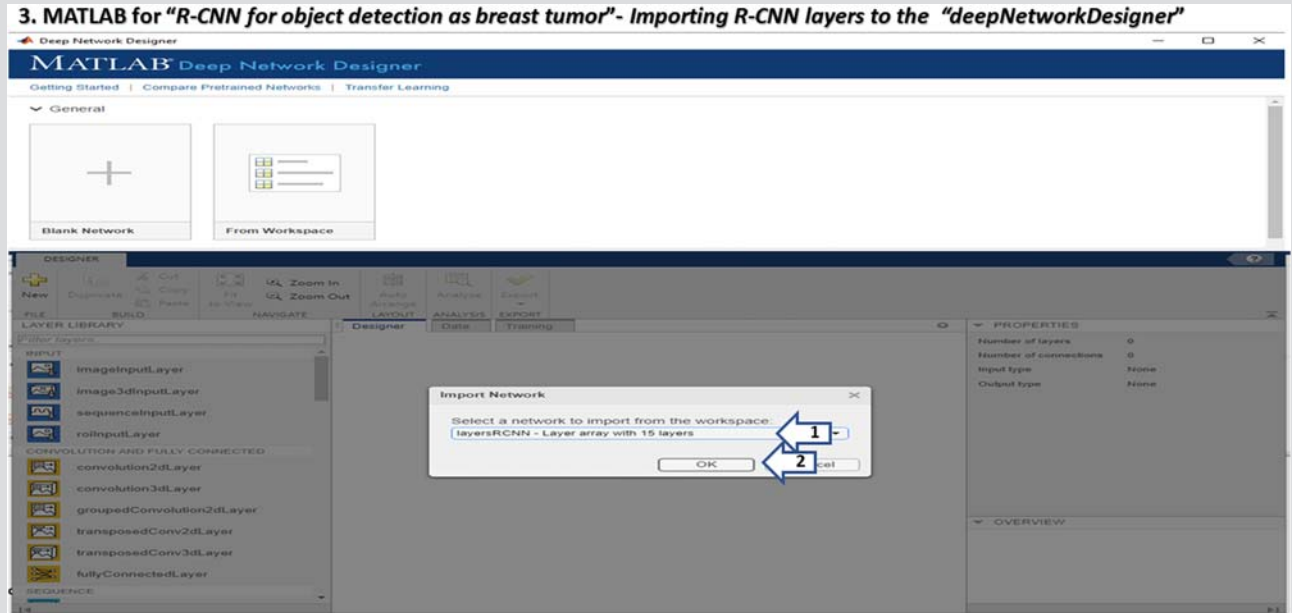
	1	2
	imageFilename	breastTumor-XYWH
1	'breastTumor/MAMMO-1.JPG'	[385,1227,212,197]
2	'breastTumor//MAMMO-2.JPG'	[1760,1508,208,224]
3	'breastTumor//MAMMO-3.JPG'	[758,968,68,92]
4	'breastTumor/MAMMO-4.JPG'	[342,938,116,150]
5	'breastTumor/MAMMO-5.JPG'	[126,674,118,100]
6	'breastTumor//MAMMO-6.JPG'	[718,1282,143,121]
7	'breastTumor/MAMMO-7.JPG'	[1764,1124,197,221]
8	'breastTumor/MAMMO-8.JPG'	[1588,867,303,218]
9	'breastTumor/MAMMO-9.JPG'	[1872,684,260,184]
10	'breastTumor//MAMMO-10.JPG'	[1848,1060,248,264]

An arrow labeled '2' points to the second column of the table.

Notes: The MATLAB variable “breastTumor” is a table 10 x 2 that has the “imageFilename” and position of the breast tumor of 2117 x 2606, and the object to detect is the “tumor breast formation” indicated with a rectangle as “XYWH” for each of the 10 images provided in the folder “breastTumorLabel”. The table can be generated by MATLAB tool in “Computer Vision Toolbox” known as “Image Labeler”. From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

3

MATLAB for “R-CNN for object detection as breast tumor” — Importing R-CNN layers to the “deepNetworkDesigner”
In the home screen of the “deepNetworkDesigner” select “workspace”, then select the variable: “layersRCNN—Layer array with 15 layers,” finally click on “OK” button



In the home screen of the “deepNetworkDesigner” select “workspace”, then select the variable: “layersRCNN – Layer array with 15 layers”, finally click on “OK” button.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 4 Description
MATLAB for “R-CNN for object detection as breast tumor”—R-CNN 15 layers and zooming into ‘imageInputLayer’
The top screen shows the complete “R-CNN with 15 layers, 14 connections for classification.” “Zoom in” to the beginning of the net to check the properties of the first layer: ‘imageInputLayer’

Screen figure

4. MATLAB for “R-CNN for object detection as breast tumor”- R-CNN: all 15 layers and zooming into ‘imageInputLayer’

The top screen shows the complete “R-CNN with 15 layers, 14 connections for classification”. “Zoom in” to the beginning of the net to check the properties of the first layer: ‘imageInputLayer’.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

5

MATLAB for “R-CNN for object detection as breast tumor”—R-CNN layers: ‘convolution2dLayer’ and ‘maxPooling2dLayer’
The top screen shows the ‘convolution2dLayer’ and its properties. The bottom screen shows the ‘maxPooling2dLayer’ and its properties

5. MATLAB for “R-CNN for object detection as breast tumor”— R-CNN layers: ‘convolution2dLayer’ and ‘maxPooling2dLayer’

The top screen shows the ‘convolution2dLayer’ and its properties. The bottom screen shows the ‘maxPooling2dLayer’ and its properties

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

6 MATLAB for “R-CNN for object detection as breast tumor”—R-CNN layers: ‘reluLayer’ and ‘convolution2dLayer’
The top screen shows the ‘reluLayer’ and its properties. The bottom screen shows the ‘convolution2dLayer’ and its properties

6. MATLAB for “R-CNN for object detection as breast tumor”— R-CNN layers: ‘reluLayer’ and ‘convolution2dLayer’

The top screenshot shows the MATLAB Designer interface with the 'reluLayer' selected. The Properties pane on the right displays the 'reluLayer' properties, including the Name field set to 'relu'. The bottom screenshot shows the same interface with the 'convolution2dLayer' selected. The Properties pane on the right displays the 'convolution2dLayer' properties, including the Name field set to 'conv_1', FilterSize set to 5, NumFilters set to 32, and Stride set to 1. Both screenshots include a Layer Library on the left and a central Designer workspace with a flowchart of layers.

The top screen shows the ‘reluLayer’ and its properties. The bottom screen shows the ‘convolution2dLayer’ and its properties

7

MATLAB for “R-CNN for object detection as breast tumor”—R-CNN layers: ‘reluLayer’ and ‘averagePooling2dLayer’
The top screen shows the ‘reluLayer’ and its properties. The bottom screen shows the ‘averagePooling2dLayer’ and its properties

7. MATLAB for “R-CNN for object detection as breast tumor”— R-CNN layers: ‘reluLayer’ and ‘averagePooling2dLayer’

The top screen shows the ‘reluLayer’ and its properties. The bottom screen shows the ‘averagePooling2dLayer’ and its properties

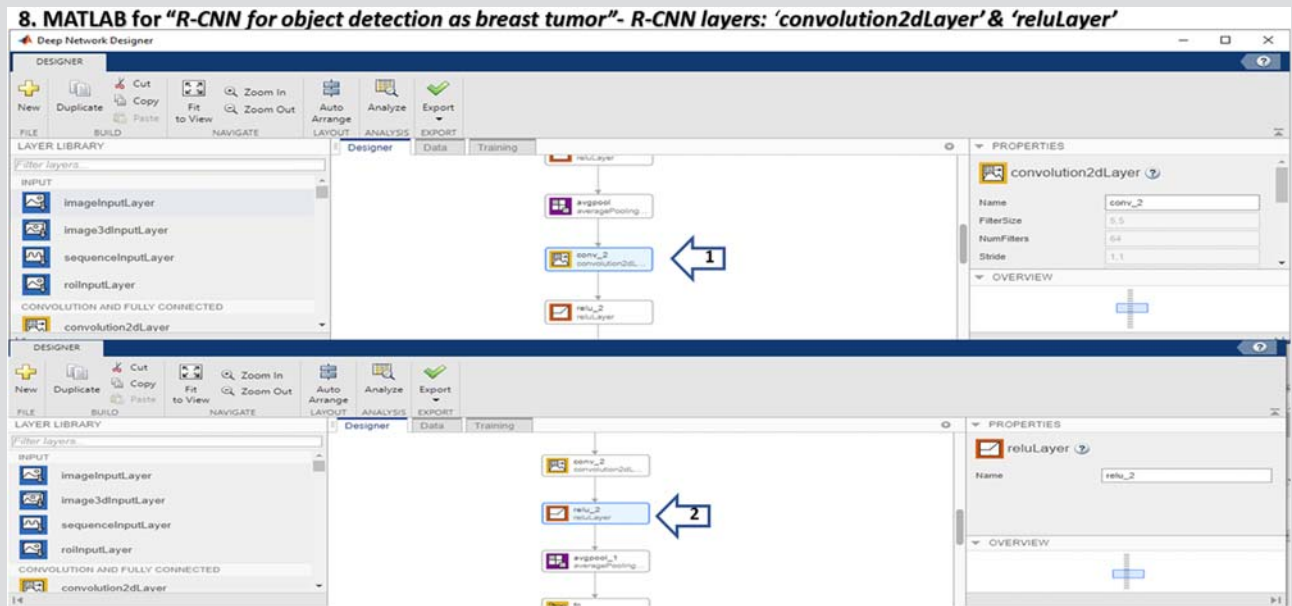
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 8 Description
MATLAB for “R-CNN for object detection as breast tumor”—R-CNN layers: ‘convolution2dLayer’ & ‘reluLayer’
The top screen shows the ‘convolution2dLayer’ and its properties. The bottom screen shows the ‘reluLayer’ and its properties

Screen figure



The top screen shows the ‘convolution2dLayer’ and its properties. The bottom screen shows the ‘reluLayer’ and its properties

9

MATLAB for “R-CNN for object detection as breast tumor”—R-CNN layers:

‘averagePooling2dLayer’ and ‘fullyConnectedLayer’

The top screen shows the ‘averagePooling2dLayer’ and its properties. The bottom screen shows the ‘fullyConnectedLayer’ and its properties

9. MATLAB for “R-CNN for object detection as breast tumor”— R-CNN layers: ‘averagePooling2dLayer’ and ‘fullyConnectedLayer’

The top screenshot shows the MATLAB Deep Network Designer interface. The central workspace displays a neural network diagram with layers: 'relu_2 ReLU Layer', 'avgpool_1 averagePooling2dLayer', 'fc fullyConnected...', and 'relu_3 ReLU Layer'. A blue arrow labeled '1' points to the 'avgpool_1' layer. The Properties window on the right shows the configuration for 'averagePooling2dLayer': Name 'avgpool_1', PoolSize '3.3', Slide '2.2', and Padding '0.0.0.0'. The bottom screenshot shows the same interface with the 'fc fullyConnected...' layer highlighted by a blue arrow labeled '2'. The Properties window on the right shows the configuration for 'fullyConnectedLayer': Name 'fc', InputSize '576', OutputSize '124', and Weights '[64+576 single]'.

The top screen shows the ‘averagePooling2dLayer’ and its properties. The bottom screen shows the ‘fullyConnectedLayer’ and its properties.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

10 MATLAB for “R-CNN for object detection as breast tumor”—R-CNN layers: ‘reluLayer’ and ‘fullyConnectedLayer’
The top screen shows the ‘reluLayer’ and its properties. The bottom screen shows the ‘fullyConnectedLayer’ and its properties

10. MATLAB for “R-CNN for object detection as breast tumor”— R-CNN layers: ‘reluLayer’ and ‘fullyConnectedLayer’

The top screen shows the ‘reluLayer’ and its properties. The bottom screen shows the ‘fullyConnectedLayer’ and its properties

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

11

MATLAB for “R-CNN for object detection as breast tumor”—R-CNN layers: ‘softMaxLayer’ and ‘classificationLayer’

The top screen shows the ‘softMaxLayer’ and its properties. The bottom screen shows the ‘classificationLayer’ and its properties, be sure that the “OutputSize = auto.” Finally, click on “Analyze” button

11. MATLAB for “R-CNN for object detection as breast tumor”- R-CNN layers: ‘softMaxLayer’ and ‘classificationLayer’

The top screen shows the ‘softMaxLayer’ and its properties. The bottom screen shows the ‘classificationLayer’ and its properties, be sure that the “OutputSize=auto” . Finally, click on “Analyze” button.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 12 Description
MATLAB for “R-CNN for object detection as breast tumor”—R-CNN layers: Net Analyzer
The “Deep Learning Network Analyzer” must indicate the status of the “15 layers,” with “0 warnings” and “0 errors.” Go back to the MATLAB workspace to continue

Screen figure

12. MATLAB for “R-CNN for object detection as breast tumor”- R-CNN layers: Net Analyzer

Name	Type	Activations	Learnables
1 imageinput 32x32x3 images with 'zerocenter' normalization	Image Input	32x32x3	-
2 conv 32 5x5x3 convolutions with stride [1 1] and padding [2 2 2 2]	Convolution	32x32x32	Weights 5x5x3x32 Bias 1x1x32
3 maxpool 3x3 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	15x15x32	-
4 relu ReLU	ReLU	15x15x32	-
5 conv_1 32 5x5x32 convolutions with stride [1 1] and padding [2 2 2 2]	Convolution	15x15x32	Weights 5x5x32x32 Bias 1x1x32
6 relu_1 ReLU	ReLU	15x15x32	-
7 avgpool 3x3 average pooling with stride [2 2] and padding [0 0 0 0]	Average Pooling	7x7x32	-
8 conv_2 64 5x5x32 convolutions with stride [1 1] and padding [2 2 2 2]	Convolution	7x7x64	Weights 5x5x32x64 Bias 1x1x64
9 relu_2 ReLU	ReLU	7x7x64	-
10 avgpool_1 3x3 average pooling with stride [2 2] and padding [0 0 0 0]	Average Pooling	3x3x64	-
11 fc 64 fully connected layer	Fully Connected	1x1x64	Weights 64x576 Bias 64x1
12 relu_3 ReLU	ReLU	1x1x64	-
13 fc_rnn 2 fully connected layer	Fully Connected	1x1x2	Weights 2x64 Bias 2x1
14 softmax softmax	Softmax	1x1x2	-
15 classoutput crossentropyex	Classification Output	-	-

The “Deep Learning Network Analyzer” must indicate the status of the “15 layers”, with “0 warnings” and “0 errors”. Go back to the MATLAB workspace to continue.

13

MATLAB for “R-CNN for object detection as breast tumor” –Step 2) Define Options for Training & Train R-CNN detector
 Copy and paste in the command prompt: “Step 2) Define Options for Training & Train R-CNN detector.” Where the “train options” are defined with its parameters, the “training for objects detection” is made, and their progress is shown as a table for the “40-epoch” processed in “one CPU in 4:29 min” with “Accuracy = 90.63%”. Note: This values depends of the processor speed on your computer

13. MATLAB for “R-CNN for object detection as breast tumor”- Step 2) Define Options for Training & Train R-CNN detector

The screenshot shows the MATLAB environment with the following components:

- Workspace:** Lists variables like 'breastTumor', 'imDir', 'layersRCNN', 'options', and 'rcnn'.
- Current Folder:** Shows the directory structure including 'breastTumor', 'breastTumorLabel', 'breastTumor.mat', 'detectBreastTumorRCNN.m', 'layersRCNN.mat', and 'MAMMO-test.JPG'.
- Editor:** Displays the MATLAB script 'detectBreastTumorRCNN.m' with the following code:


```

11 % Step 2) Define Options for Training & Train R-CNN detector.
12 options = trainingOptions('sgdm', ...
13 'MiniBatchSize', 32, ...
14 'InitialLearnRate', 1e-6, ...
15 'MaxEpochs', 40);
16 %The nextfunction requires Deep Learning Toolbox™ & Statistics and Machine Learning Toolbox™.
17 rcnn = trainRCNNObjectDetector(breastTumor, layersRCNN, options,'NegativeOverlapRange', [0 0.5]);
      
```

 A blue arrow labeled '1' points to the 'options' definition.
- Command Window:** Shows the execution progress:


```

- BREASTTUMORRCNN
--> Extracting region proposals from 10 training images...done.
--> Training a neural network to classify objects in training data...
Training on single CPU.
Initializing input data normalization.
=====
| Epoch | Iteration | Time Elapsed | Mini-batch | Mini-batch | Base Learning |
|       |          | (hh:mm:ss)  | Accuracy  | Loss       | Rate          |
|-----|-----|-----|-----|-----|-----|
| 1 | 1 | 00:00:00 | 75.00% | 3.8831 | 1.0000e-06 |
| 7 | 80 | 00:00:42 | 75.00% | 2.4753 | 1.0000e-06 |
| 13 | 100 | 00:01:26 | 75.00% | 1.4380 | 1.0000e-06 |
| 19 | 150 | 00:02:10 | 75.00% | 1.4717 | 1.0000e-06 |
| 25 | 200 | 00:02:50 | 81.25% | 0.7638 | 1.0000e-06 |
| 32 | 250 | 00:03:30 | 81.25% | 1.1089 | 1.0000e-06 |
| 38 | 300 | 00:04:14 | 81.25% | 0.3074 | 1.0000e-06 |
| 40 | 320 | 00:04:29 | 90.63% | 0.3872 | 1.0000e-06 |
=====
Network training complete.
--> Training bounding box regression models for each object class...100.00%...done.
Detector training complete.
      
```

 A blue arrow labeled '2' points to the table.

Copy and paste in the command prompt: “Step 2) Define Options for Training & Train R-CNN detector”. Where the “train options” are defined with its parameters, the “training for objects detection” is made, and their progress is shown as a table for the “40-epoch” processed in “one CPU in 4:29 min” with “Accuracy =90.63%”. Note: This values depends of the processor speed on your computer

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

14 MATLAB “R-CNN for object detection as breast tumor”—Step 3) Testing R-CNN detector for breast tumors in mammogram Copy and paste in the command prompt: “Step 3) Testing R-CNN detector for Detect objects using Fast R-CNN.” The image to test is loaded, and the variables for the best detection are calculated: ‘bbox’ (object coordinates XYWH), ‘score’ (best probabilities), and ‘label’ (labels for best probability)

14. MATLAB “R-CNN for object detection as breast tumor” - Step 3) Testing R-CNN detector for breast tumors in mammogram

Image for object detection using R-CNN

```
>> img = imread('MAMMO-test.jpg'); % Load image object detection breast tumor
figure: imshow(img);title("Image for object deection using R-CNN")
>> figure: imshow(img);title("Image for object deection using R-CNN")
>> [bbox, score, label] = detect(rcnn,img,'MiniBatchSize',32); % Detect box,score and label
```

bbox				score		label	
2x4 double				2x1 single		2x1 categorical	
	1	2	3	4			
1	1163	290	19	38	0.7393	1	breastTumorXYWH
2	1607	1162	276	333	0.9753	2	breastTumorYYWH

Copy and paste in the command prompt: “Step 3) Testing R-CNN detector for Detect objects using Fast R-CNN”. The image to test is loaded, and the variables for the best detection are calculated: ‘bbox’ (object coordinates XYWH), ‘score’ (best probabilities), and ‘label’ (labels for best probability).

MATLAB for “R-CNN for object detection as breast tumor”—Step 4) Display image with object detection with best probability or score

Copy and paste in the command prompt: “Step 4) Display image with object detection with best probability.” The mammogram image is added with the insertion of the object annotation: ‘bbox’ (object coordinates XYWH), ‘score’ (best probabilities), and ‘label’ (labels for best probability). Finally, the “breastTumor” directory add at the beginning to the path is remove from the Matlab path

15. MATLAB for “R-CNN for object detection as breast tumor”— Step 4) Display image with object detection with best probability

The screenshot shows the MATLAB environment with the following components:

- Workspace:** Lists variables such as 'annotation', 'bbox', 'breastTumor', 'detectBreastTumor', 'idx', 'imgDir', 'img', 'label', 'layersRCNN', 'options', 'rcnn', and 'score'.
- Editor:** Contains the script 'detectBreastTumorRCNN.m' with the following code:


```

24 %% Step 4) Display image with object detection with best probability
25 [score, idx] = max(score); % Find maximum score
26 bbox = bbox(idx, :); % Rectangle position of maximum score
27 annotation = sprintf('%s (Probability = %f)', 'Breast Tumor', score);
28 detectBreastTumor = insertObjectAnnotation(img, 'rectangle', bbox, annotation);
29 figure: imshow(detectBreastTumor);title("Processed Mammogram with tumor breast detection");
30 rmpath(imgDir); % Remove the breastTumor directory from Matlab path
      
```
- Command Window:** Shows the execution of the script:


```

>> %% Step 4) Display image with object detection with best probability
[score, idx] = max(score); % Find maximum score
bbox = bbox(idx, :); % Rectangle position of maximum score
annotation = sprintf('%s (Probability = %f)', 'Breast Tumor', score);
detectBreastTumor = insertObjectAnnotation(img, 'rectangle', bbox, annotation);
figure: imshow(detectBreastTumor);
rmpath(imgDir); % Remove the breastTumor directory from Matlab path
imshow(detectBreastTumor);title("Processed image with object detection");
figure: imshow(detectBreastTumor);title("Processed Mammogram with tumor breast detection");
      
```
- Figure:** A mammogram image titled "Processed Mammogram with tumor breast detection" with a yellow bounding box around a detected tumor. A label above the box reads "Breast Tumor (Probability = 0.975277)".

Copy and paste in the command prompt: “Step 4) Display image with object detection with best probability”. The mammogram image is added with the insertion of the object annotation: ‘bbox’ (object coordinates XYWH), ‘score’ (best probabilities), and ‘label’ (labels for best probability). Finally, the breastTumor directory add is remove from the Matlab path.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

“*R-CNN model from MATLAB Deep Learning Toolbox classification for object detection of breast tumor in mammogram*” [17], allows obtaining an AI model to detect objects and classify them as assigned to their categories or classes.

Recommendation

This kind of “*R-CNN model*” can be applied to a diversity of dataset to detect special areas in images in different topics in biomedical engineering.

6.2.6 Hopfield Network

“*Hopfield Network (HN)*” is a form of “*recurrent artificial neural network*” that can reconstruct data after being fed with corrupt versions of the same data. It usually works by “*first learning several binary patterns and then returning the one that is the most like a given input.*” “*Hopfield networks*” also provide a model for understanding human memory; they can be described as a network of nodes, representing neurons that are connected by links, each unit has one of two states at any point in time and vectors/matrices representing the state of each node. The links represent the connections between nodes, and they are symmetrical. The specific purpose of the “*HN*” is to store one or more patterns and then recall the full pattern based on partial input from them as “*associative*

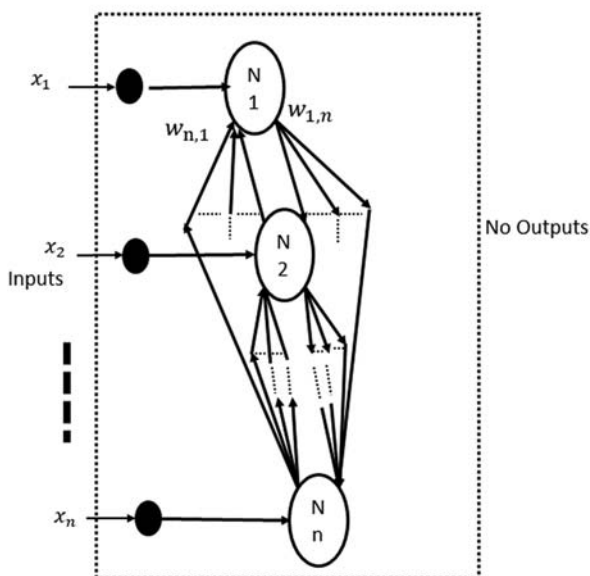
memory.” This is made using a special net architecture where all of its network is fully interconnected nodes considered as input to a single visible layer and no outputs, as shown in Fig. 6.7A. The units in the nodes “*Hopfield networks*” have binary threshold units from “*1 to -1*” or “*1 to 0,*” where the connection has been specified under the following restrictions: (1) $w_{i,i} = 0$ meaning that no node has a connection with itself, and (2) $w_{i,j} = w_{j,i}$ meaning that connections are symmetrical. The “*weight matrix*” differentiates the behavior of an one Hopfield network from another in a recursive way. The updating in the binary states z_i in its nodes are defined as:
$$z_i = \begin{cases} +1 & \text{if } \sum_j w_{i,j} N_j \geq \theta_i \\ -1 & \text{otherwise} \end{cases}$$
, and the specific “*HN*” has a value named “*Energy (E)*” that represents the total state of the network, represented as: $E = -\frac{1}{2} \sum_{i,j} w_{i,j} N_i N_j + \sum_i \theta_i N_i$, as indicated in Fig. 6.7B. The updating or correction can be “*Asynchronous,*” where only one unit it is updated at a time, or “*Synchronous,*” where all units are updated at the same time. Under repeated updating, the “*HN*” will eventually converge to a state which is the local minimum in the energy function.

Some currently used examples of “*Hopfield Network (HN)*” in “*Biomedical Engineering*” are:

- “*Identification of noisy dynamical systems with parameter estimation based on Hopfield neural networks*” by

Hopfield Network (HN)**(A) HN architecture**

HN is a fully connected nodes, no inputs or outputs

**(B) HN equations**

HN Node Restrictions:

- 1) $w_{i,i} = 0$ no node has a connection with itself
- 2) $w_{i,j} = w_{j,i}$ connections are symmetric

Where: $w_{i,j}$ is the node strength connection between i,j

HN uses “Binary threshold” to make Node Update:

- 3) **Binary state**
$$z_i = \begin{cases} +1 & \text{if } \sum_j w_{i,j} N_j \geq \theta_i \\ -1 & \text{otherwise} \end{cases}$$

Where: N_i = state of node i , θ_i = threshold i

HN total Energy (E):

- 4)
$$E = -\frac{1}{2} \sum_{i,j} w_{i,j} N_i N_j + \sum_i \theta_i N_i$$

FIGURE 6.7 Hopfield Network (HN): (A) HN architecture, and (B) HN equations.

Atencia et al. [74], where an algorithm for estimating time-varying parameters of dynamical systems is proposed within the large family of prediction error methods. The algorithm is based on the ability of “Hopfield neural networks” to solve optimization problems, since its formulation can be summarized as minimization of the prediction error by means of a “continuous Hopfield network” that eventually destabilizes a closed control loop.

- “Human fringe skeleton extraction by an improved Hopfield neural network with direction features” by Huang et al. [75], where the fringe skeleton is a useful means of shape description of three-dimensional (3D) human bodies. A new method is proposed to extract the human fringe skeleton by matching feature points of the torso and tracking on the direction of the limb. They primarily extract the depth direction feature based on the coronal plane of a 3D human model. By introducing three transverse cross sections on the torso and the concept of triangle pairs, the extraction result of the coronal plane and its depth direction feature will not be affected by shading and interference of the limb on different postures. With the local direction feature, an improved Hopfield neural network (IMHNN) is used to locate feature points of any position by matching feature points of the template model and the personalized target model.

The “Hopfield network” is unsupervised and it is commonly used for autoassociation memorizing paths, denoising inputs images, and optimization tasks. “HN” can be used in many biomedical applications, such as modeling brain disorders like “Traumatic Brain Disorders,” analyzing brain’s physical damage in MRI images, memorizing genes expressions [18], detecting “NLP” for text paths, “images paths” applications, and many others.

6.2.6.1 Research 6.3 Hopfield Network model to reconstruct noisy chest X-ray images

6.2.6.1.1 Case for research

“Obtain an HN model from MATLAB Deep Learning Toolbox to reconstruct noisy chest X-ray images.”

6.2.6.1.2 General objective

Apply “MATLAB Deep Learning Toolbox” to define, build, and train an “HN Neural Network model to reconstruct noisy chest X-ray images.”

6.2.6.1.3 Specific objectives

- Read training chest X-ray images dataset.
- Train all chest X-ray images stored in the train directory.

- Load a test image stored at the test directory.
- Used HM model to compare test with trained images and reconstruct it.

6.2.6.1.4 Background for “chest X-ray images”

“Chest X-rays” show images of heart, lungs, blood vessels, airways, and the bones of the chest and spine. The image helps a doctor determine whether you have heart problems, a collapsed lung, pneumonia, broken ribs, emphysema, cancer, or any of several other conditions. Some people have a series of chest X-rays done over time to track whether a health problem is getting better or worse. A “chest X-ray” can reveal [19]:

- “Lungs conditions,” such as detect of cancer, infection, or air collecting in the space around a lung, which can cause the lung to collapse. They can also show chronic lung conditions, such as emphysema or cystic fibrosis.
- “Heart-related lung problems” can show changes or problems in lungs that stem from heart problems, such as fluid in lungs can be a result of “congestive heart failure.”
- “Size and outline of heart,” which may indicate a heart failure.
- “Blood vessels,” such as large vessels near the heart: aorta and pulmonary arteries and veins, which may reveal aortic aneurysms, other blood vessel problems, or congenital heart disease.
- “Calcium deposits in heart and vessels” may indicate fats and other substances in your vessels, which can damage heart valves, coronary arteries, heart muscle, or the protective sac that surrounds the heart. Calcified nodules in lungs are most often from an old, resolved infection.
- “Fractures in Rib or Spine fractures or other problems with bones.”
- “Postoperative chest changes.”

The most frequently requested chest X-ray images are posterior-anterior (PA) X-ray or the PA X-ray, and the lateral chest X-ray. Others are the decubitus view, the lordotic view, the inspiration/expiration series, chest fluoroscopy, dual-energy chest X-ray, digital tomography, or tomosynthesis.

6.2.6.1.5 Dataset

The dataset consists of two folders, as shown in Fig. 6.8: “Train” with three normal chest X-rays, and “Test” with one noised normal chest X-rays. The objective is to prove that the reconstruction of noised images using the “HN Neural Network model” is possible by comparing the test image with the trained images.

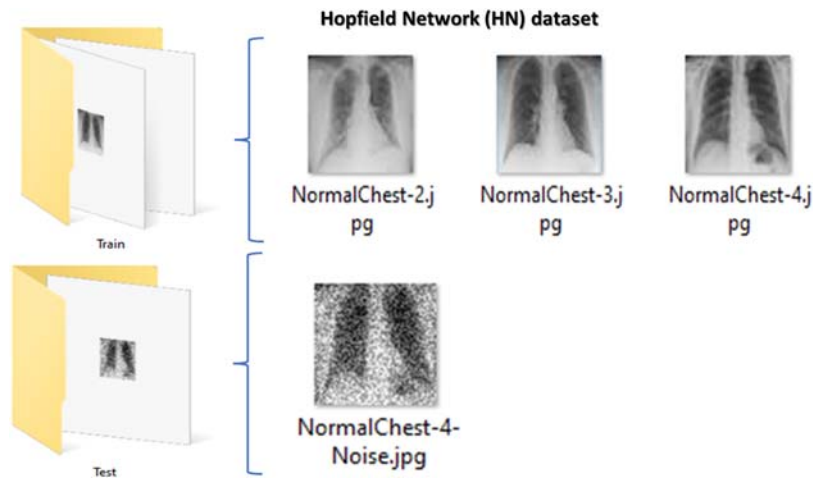


FIGURE 6.8 The dataset for the Neural Network model to reconstruct the noisy chest X-ray images dataset.

Note: This images dataset is available in the companion directory of the book, in the following folder “`..\Exercises_book_ABME\CH6\MATLAB_HN.`”

6.2.6.1.6 Procedure

The steps to obtain an “*HN model from MATLAB Deep Learning Toolbox reconstruct noisy chest X-ray images*”

are summarized in *Table of slides 6.3*, and each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

Table of slides 6.3 Steps for MATLAB Deep Learning Toolbox to obtain HN model from MATLAB Deep Learning Toolbox to reconstruct noisy chest X-ray images.”

Slide Description

Screen figure

1 Open MATLAB “Hopfield Network to be used in noisy chest X-rays”—Step 1) Read Training chest X-ray images dataset
Go to the directory “... \Exercises_book_ABME\CH6\MATLAB_HN.” Open the script “HNf.m.” Copy and paste in the command prompt: “Step 1) Read Training chest X-ray images dataset.” Where the original images are read as 64 × 64 pixels RGB images, then converted to binary format and store in the cell variable “Output”

1. Open MATLAB “Hopfield Network to be used in noisy chest x-rays”- Step 1) Read Training chest x-rays images dataset

The screenshot shows the MATLAB IDE with the following components:

- Workspace:** Lists variables: `f` (3x1 struct), `files` (1x3 cell), `lm` (1x3 cell), `k` (3), `n` (3), `Output` (1x3 cell), and `str` (1x3 cell). A blue arrow points to the `Output` variable.
- Current Folder:** Shows the directory structure: `other`, `Test`, `Train`, `HNf.asv`, and `HNf.m`.
- Editor:** Displays the script `HNf.m` with the following code:


```

1 % Chap. 5 APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
2 % Elsevier - Academic Press by: Jorge Garza-Ulloa
3
4 %% Step1) Read Training chest x-rays images dataset
5 % All Chest x-ray images must have pixel squared res. 64 x 64
6 od Train=fdir('*.jpg');files=(f.name);n=numel(files);
7 - for k=1:n
8   Im(k)=imread(files(k)); % Read grey image
9   Im(k)=imresize(Im(k), [64 NaN]); % resize image to 64 x 64
10  Output(k) = im2bw(Im(k)); % Output in Black & White
11  Output(k) = im2double(Output(k));% Output in double number
12 - end
13 - figure;
14 - for i = 1:n; subplot(1,n,i);imshow(Im(i));
15 -   str=sprintf('Input RGB=%d',i);title(str);
16 - end
17 - od ..
            
```

 A blue arrow points to line 11.
- Command Window:** Shows the execution of the script:


```

im(k)=imresize(im(k), [64 NaN]);
Output(k) = im2bw(Im(k)); % Output in Black & White
Output(k) = im2double(Output(k));% Out double number
end
figure;
for i = 1:n; subplot(1,n,i);imshow(Im(i));
str=sprintf('Input RGB=%d',i);title(str);
end
            
```

 A blue arrow points to the `Output(k) = im2double(Output(k));` line.
- Figure:** Displays three chest x-ray images labeled `Input RGB=1`, `Input RGB=2`, and `Input RGB=3`. A blue arrow points to the text `Training chest x-rays images dataset` below the images.

Go to the directory “... \Exercises_book_ABME\CH6\MATLAB_HN”. Open the script “HNf.m”. Copy and paste in the command prompt: “Step 1) Read Training chest x-rays images dataset”. Where the original images are read as 64 x 64 pixels RGB images, then converted to binary format and store in the cell variable “Output”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

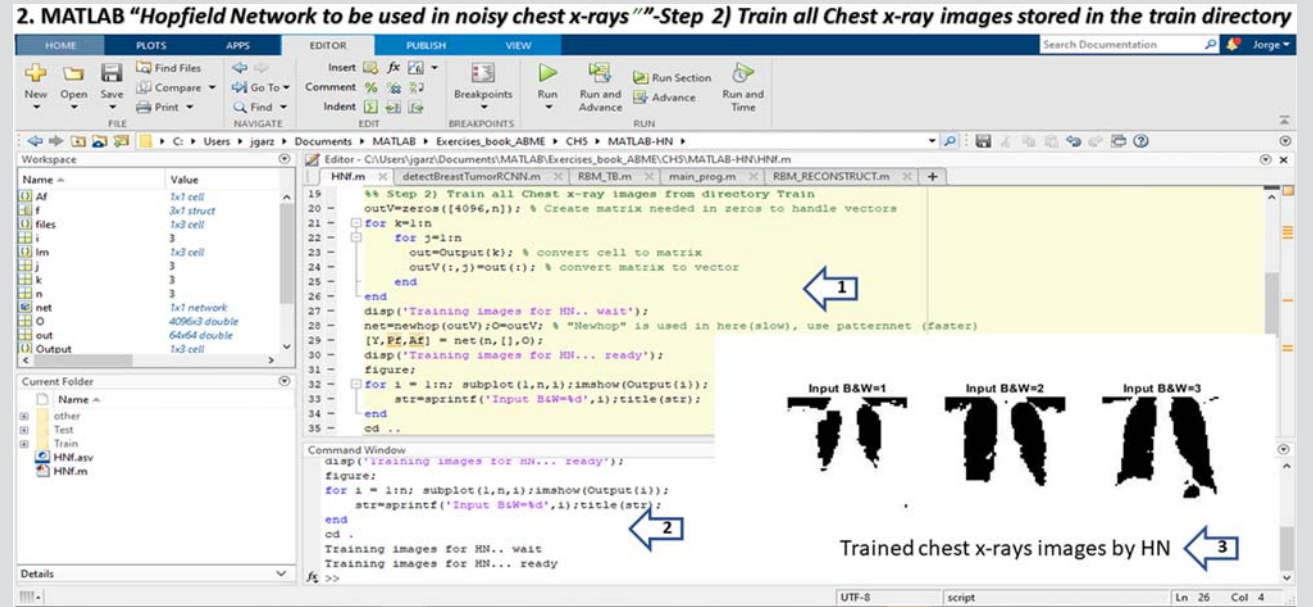
(Continued)

(Continued)

Slide Description

Screen figure

2 MATLAB "Hopfield Network to be used in noisy chest X-rays"—
Step 2) Train all Chest X-ray images stored in the "train directory"
Copy and paste in the command prompt: "*Step 2) Train all Chest X-ray images from directory Train.*"
For this example, the MATLAB function "*newhop()*" is used, but *patternnet()* is recommended as a newer more efficient function. Therefore all images were resized to "*64 × 64 pixels for a total of 4096 values,*" which are to be handled for each image by the "*HN*"



Copy and paste in the command prompt: "*Step 2) Train all Chest x-ray images from directory Train.*" For this example, the MATLAB function "*newhop()*" is used, but *patternnet()* is recommended as a newer more efficient function. Therefore, all images were resized to "*64 x 64 pixels for a total of 4096 values,*" that are to be handled for each image by the "*HN*".

3

MATLAB “Hopfield Network used in noisy chest X-rays”—Step 3) Load a test image stored at the test directory
Copy and paste in the command prompt: “Step 3) Load test image from directory Test.” Only the first image stored is read, and it is shown as “Test Input RBC = 1” and its conversion to binary as “Test Input B&W = 1”

3. MATLAB “Hopfield Network used in noisy chest x-rays”- Step 3) Load a test image stored at the test directory

The screenshot shows the MATLAB IDE with the following code in the Editor window:

```
36 %% Step 3) Load test image from directory Test
37 % One Chest x-ray images with noise pixel squared res. 64 x 64 in dir Test
38 - od Test;
39 - f=dir('*.jpg');files={f.name};m=1; % only one image to test
40 - for k=1:m
41 -     Im(k)=imread(files(k));
42 -     Im(k)=imresize(Im(k), [64 NaN]); % resize image to 64 x 64
43 -     OutputT(k) = im2bw(Im(k)); %Output = im2bw(Output)
44 -     OutputT(k) = im2double(OutputT(k));
45 - end
46 - figure;
47 - for i = 1:m;
48 -     subplot(2,2,1);imshow(Im(i));str=sprintf('Test Input %d',i);
49 -     subplot(2,2,2);imshow(OutputT(i));str=sprintf('Test Input %d',i);
50 - end
51 - od ..
```

The Command Window shows the following output:

```
OutputT(k) = im2bw(Im(k)); %Output = im2bw(Output)
OutputT(k) = im2double(OutputT(k));
end
figure;
for i = 1:m;
subplot(2,2,1);imshow(Im(i));str=sprintf('Test Input RBG=%d',i);
subplot(2,2,2);imshow(OutputT(i));str=sprintf('Test Input B&W=%d',i);
end
od ..
fs >>
```

Two subplots are displayed: "Test Input RBG=1" showing a noisy chest X-ray image, and "Test Input B&W=1" showing the binary (black and white) version of the same image. The Command Window output is annotated with arrows pointing to the corresponding code lines and subplots.

Copy and paste in the command prompt: “Step 3) Load test image from directory Test.” Only the first image stored is read, and its is shown in a figure as “Test Input RBG=1” and its conversion to binary as “Test Input B&W=1”

(Continued)

Slide Description

Screen figure

4 MATLAB "Hopfield Network used in noisy chest X-rays" – Step 4) HM used to compare test with trained images and reconstruct it. Copy and paste in the command prompt: "Step 4) HM compare test with trained images and reconstruct it." Where the test image is fixed by the "HN model" used to trained images at Step 2), and it shows the resulting "reconstructed chest X-ray image." Close all

4. MATLAB "Hopfield Network used in noisy chest x-rays"- Step 4) Used HM to compare test with trained images & reconstruct it

The screenshot displays the MATLAB environment. The Editor window shows the following code:

```
%% Step 4) HM compare Test with trained images and reconstruct
A1=cell2mat(OutputT);
outC=A1(:); % convert matrix to vector
[Yc,Pf,Af] = net([1 5],[],outC);
B = [Yc(:)]; % convert cell to matrix
C= reshape(B(:), 64, []).'; % convert vector to matrix
figure;
subplot(2,2,1);imshow(OutputT(1));str=sprintf('Test Input BiW=%d',1);title(str);
subplot(2,2,2);imshow(C');str=sprintf('Reconstructed =%d',1);title(str);
```

The Command Window shows the execution of the code, resulting in two side-by-side grayscale images:

- Test Input B&W=1: A noisy chest X-ray image.
- Reconstructed =1: A reconstructed chest X-ray image, which is significantly clearer than the input.

Arrows labeled 1, 2, and 3 point to the code line for converting the vector to a matrix, the 'Test Input' image, and the 'Reconstructed' image, respectively.

Copy and paste in the command prompt: "Step 4) HM compare test with trained images and reconstruct it". Where the test image is fixed by the "HN model" used to trained images at step 2), and it shows the resulting "reconstructed chest x-ray image". Close all.

Conclusions

"HN model obtained from MATLAB Deep Learning Toolbox allows the reconstruction of noisy chest X-ray images," comparing the test image with the trained images.

Recommendation

This kind of "HN model" can be used to store "images paths," which can be used in many biomedical engineering applications to reconstruct and compare noisy images.

6.2.7 Boltzmann Machine

"Boltzmann Machine (BM)" is a type of "stochastic recurrent neural network and Markov random field." "BM" can be seen as the stochastic, generative counterpart of "Hopfield networks." "BM" is generative, this means it does not expect input data, it generates data, that is, it is capable of learning internal representation and it is able to represent and can solve difficult combinatorial problems. "BM," also known as "Energy-Based Models (EBM)," is an unsupervised model that involves learning a probability distribution from an original dataset and using it to make inferences about never seen data. "BM" has inputs of visible layers and one or several hidden layers, where everything is connected to everything without output layers. All the nodes are connected to all other input visible nodes or hidden nodes; this allows them to share information among themselves and self-generate

subsequent data, as shown in Fig. 6.9A. "BM" using "Stochastic Gradient Descent" only learns direct patterns, producing binary results. Unlike Hopfield nets, Boltzmann machine units take stochastic decisions about being "on" or "off." The units in "Boltzmann nets" are binary threshold units from "1 to -1" or "1 to 0," where its connection has been specified under the following restrictions, as shown in Fig. 6.9B (1) $w_{i,i} = 0$ meaning that no node has a connection with itself, and (2) $w_{i,j} = w_{j,i}$ meaning that connections are symmetric. The updating in the binary states z_i of its nodes is defined as:

$z_i = \begin{pmatrix} +1 & b_i + \sum_j w_{i,j}N_j \\ -1 & \text{otherwise} \end{pmatrix}$, then the probability of the node j when is "+1" is calculated as: $\text{prob} = \frac{1}{1+e^{-z_i}}$, and the specific "BM" Global Energy (E) that represents the total state of the network is represented as: $E = -(\sum_{i<j} w_{i,j}N_iN_j + \sum_i b_iN_i)$, as indicated in Fig. 6.9B (4). The energies of the state vectors represent how bad is the actual states values, which then is treated as an optimization problem. The stochastic dynamics of "BM" search for a good solution far away from the poor local optima.

"BM" has many applications in biomedical engineering with special emphasis in "Cognitive Science" AI models (Continued)

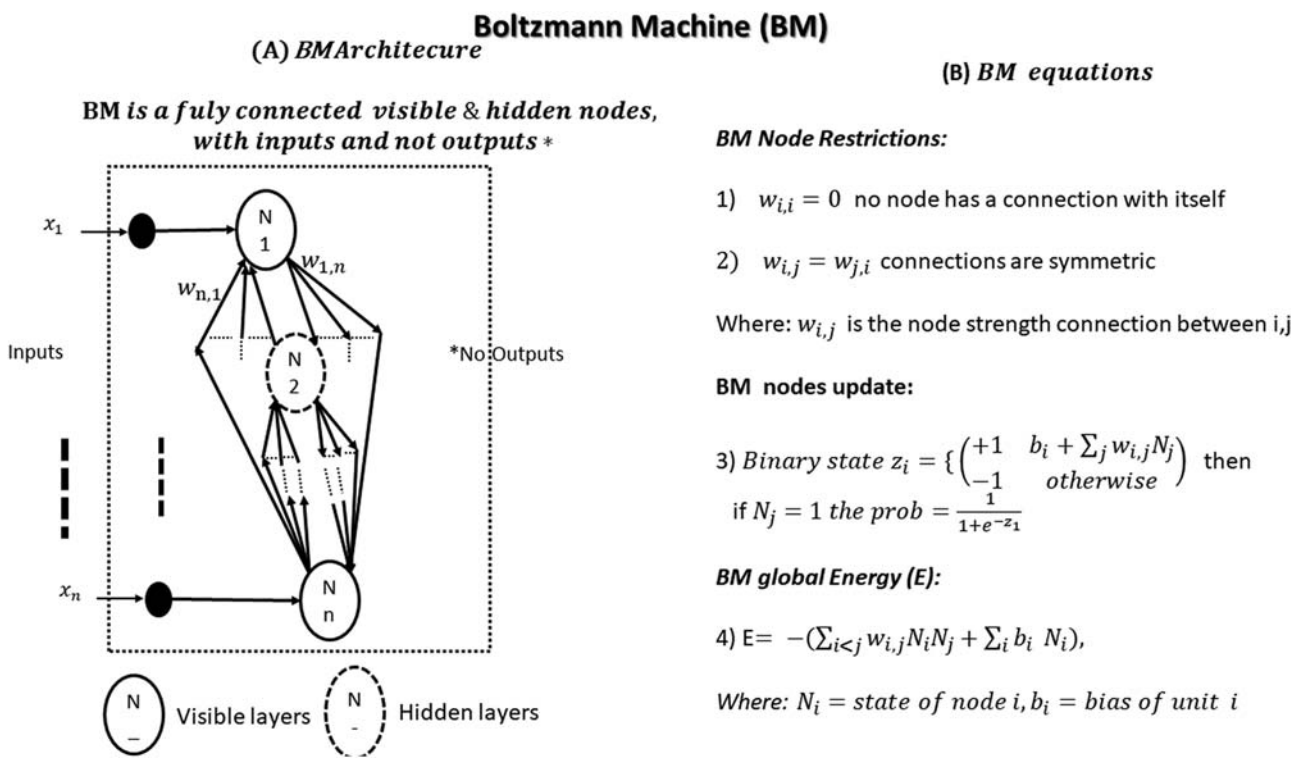


FIGURE 6.9 Boltzmann Machine (BM) NN: (A) BM architecture and (B) BM equations.

(Continued)

that will be studied in Chapter 7, section 7.6.1 Cognitive Learning and Reasoning Models Applied to Biomedical Engineering.

6.2.8 Restricted Boltzmann Machine

“Restricted Boltzmann Machine (RBM)” is a variant of a “Boltzmann Machine (BM)” that is “restricted in terms of the connections between the visible and the hidden nodes.” “RBM” is a two-layered neural network, where each visible layer node is connected to all hidden layer nodes, but with the follow restriction: “no two nodes in the same layer are connected,” as shown in Fig. 6.10A. “RBM” is a “generative stochastic artificial neural network” that can “learn a probability distribution” over its set of inputs allowing efficient algorithms, such as the “gradient-based contractive.” The units in “Restricted Boltzmann Machine net” are also binary threshold units from “1 to -1” or “1 to 0,” where the connection has been specified under the following restrictions, as shown in Fig. 6.10B: (1) non two nodes in the same layer are connected between them; (2) $w_{i,i} = 0$ meaning that no node has a connection with itself; and (3) $w_{i,j} = w_{j,i}$, meaning that connections are symmetric. There is updating in

the binary states if its nodes are defined as:

$$z_i = \begin{pmatrix} +1 & b_i + \sum_j w_{i,j}N_j \\ -1 & \text{otherwise} \end{pmatrix} \text{ in visible layer node, or}$$

$$z_j = \begin{pmatrix} +1 & b_j + \sum_i w_{i,j}H_i \\ -1 & \text{otherwise} \end{pmatrix} \text{ in hidden layer node}$$

Then if the probability of $N_i = 1$ in a visible layer node then $\text{prob} = \frac{1}{1 + e^{-z_i}} = \frac{e^{z_i}}{1 + e^{z_i}}$ or

If $H_j = 1$ in a hidden layer node then $\text{prob} = \frac{1}{1 + e^{-z_j}} = \frac{e^{z_j}}{1 + e^{z_j}}$

The specific “RBM Energy (E) is a pair of Boolean vectors” represented as:

$$E(N, H) = - \left(\sum_i b_i N_i + \sum_j b_j N_j + \sum_i \sum_j N_i w_{i,j} H_j \right), \text{ as}$$

indicated in Fig. 6.10B.

Some “RBM” example used in “Biomedical Engineering” are:

- “Restricted Boltzmann Machine method for dimensionality reduction of large spectroscopic data” by Vrabel et al. [76], is based on multivariate data obtained using “Laser-Induced Breakdown Spectroscopy (LIBS).” Dimension reduction and visualization of large datasets is a task of significant interest in spectroscopic data

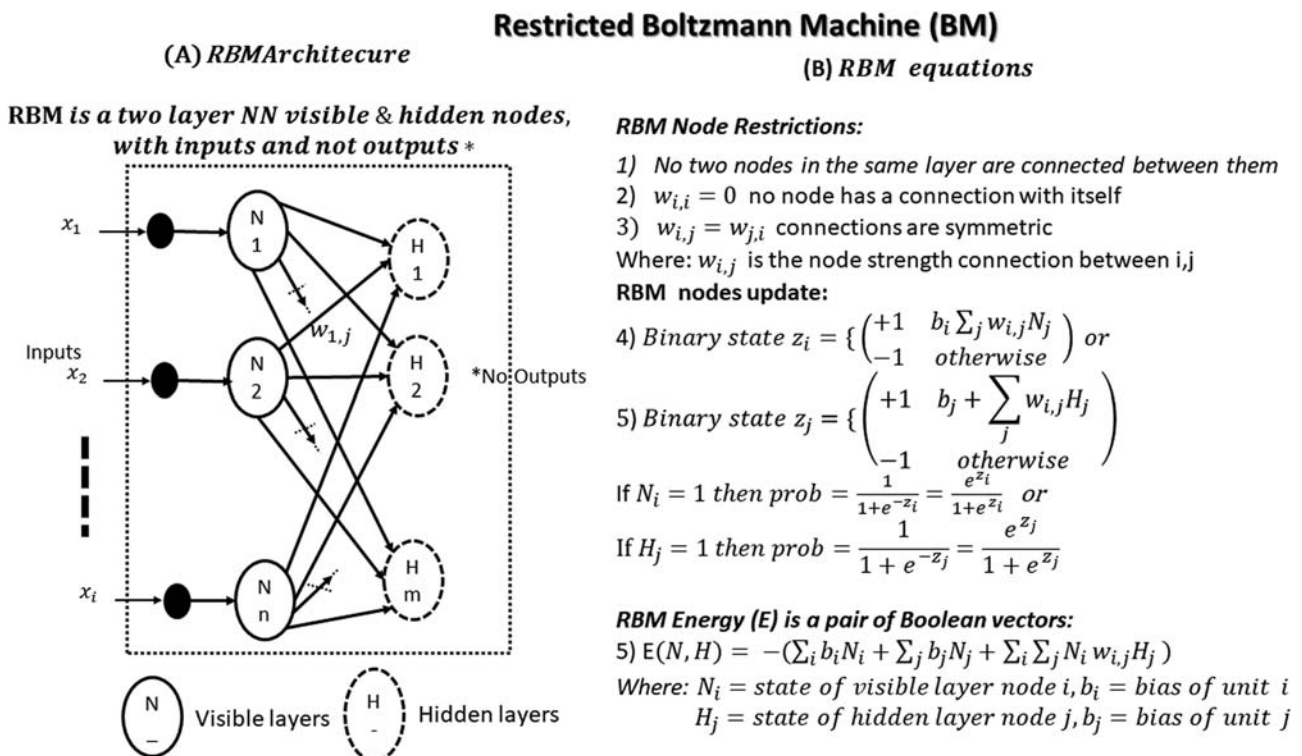


FIGURE 6.10 Restricted Boltzmann Machine (BM): (A) RBM architecture and (B) RBM equations.

processing. They propose a new methodology based on a “*Restricted Boltzmann Machine*” for dimensionality reduction of spectroscopic data and compare it to standard “*Principal Component Analysis (PCA)*.”

- “*Barrett’s esophagus analysis using infinity Restricted Boltzmann Machines*” by Passos et al. [20]. The number of patients with “*Barret’s esophagus (BE)*” has increased in recent decades; they introduce the “*infinity Restricted Boltzmann Machines (iRBMs)*” for the task of automatic identification of “*Barrett’s esophagus*” from endoscopic images of the lower esophagus. Moreover, since “*iRBM*” requires a proper selection of its *meta*-parameters, they also present a discriminative “*iRBM fine-tuning*” using six *meta*-heuristic optimization techniques.
- “*Contractive Slab and Spike Convolutional Deep Boltzmann Machine*” by Xiaojun and Haibo [21]. They propose: first, a model that extends convolution operation to the “*DBM*” to deal with real-size images. Second, they induce element-wise multiplication between real-valued slab hidden units and binary spike hidden units to enhance the quality of feature extraction in the receptive field. Then, they add the “*frobenius norm of the jacobian*” of the features as a regularization term to the maximum likelihood function to enhance the robustness of the features during training. The proposed regularization term results in a localized space contraction, which in turn obtains robust features on the hidden layer. Finally, they use a new “*block-Contractive Slab and Spike Convolutional Restricted Boltzmann Machine*” to pretrain the proposed model.

“*RBM*” can be trained in either supervised or unsupervised ways, depending on the task, and can be used for dimensionality reduction, classification, regression, collaborative filtering, feature learning, and topic modeling.

6.2.8.1 Research 6.4 Restricted Boltzmann Machine model to reconstruct noisy chest X-ray images

6.2.8.1.1 Case for research

“*Obtain an RBM model from MATLAB Deep Learning Toolbox to reconstruct noisy chest X-ray images and*

compare performance and results obtained from the HN in Research 6.3 section 6.2.6.1 Hopfield Network model to reconstruct noisy chest X-ray images.”

6.2.8.1.2 General objective

Apply “*MATLAB Deep Learning Toolbox*” to define, build, and train an “*RBM Neural Network model to reconstruct noisy chest X-ray images and compare performance and results obtained from the HN in Research 6.3 HN model.*”

6.2.8.1.3 Specific objectives

- Read training chest X-ray images dataset.
- Train all chest X-ray images stored in the train directory.
- Load a test image stored from the test directory.
- Apply RBM compares test with trained images and reconstruct it.
- Compare the results from using “*HN*” and “*RBM*” models on reconstruction and processing time.

6.2.8.1.4 Background for “chest X-ray images”

Note: Please read Research 6.3 background for “*chest X-ray images.*”

6.2.8.1.5 Dataset

The dataset consists of two folders*, as shown in Fig. 6.11: “*Train*” with three normal chest X-rays, and “*Test*” with one noised normal chest X-ray. Note*: This is a copy of the “*HN*” dataset of research 6.3. The objective is to reconstruct noised image in the test image folder using “*RBM Neural Network model*” based on trained images in the train folder, and compare the results with the ones obtained from the “*HN*” in Research 6.3.

Note: This images dataset is available in the companion directory of the book, in the following folder “*.. \Exercises_book_ABME\CH6\MATLAB_RBM.*”

6.2.8.1.6 Procedure

The steps to obtain an “*RBM model from MATLAB Deep Learning Toolbox reconstruct noisy chest X-ray images*” are summarized in *Table of slides 6.4*, and each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

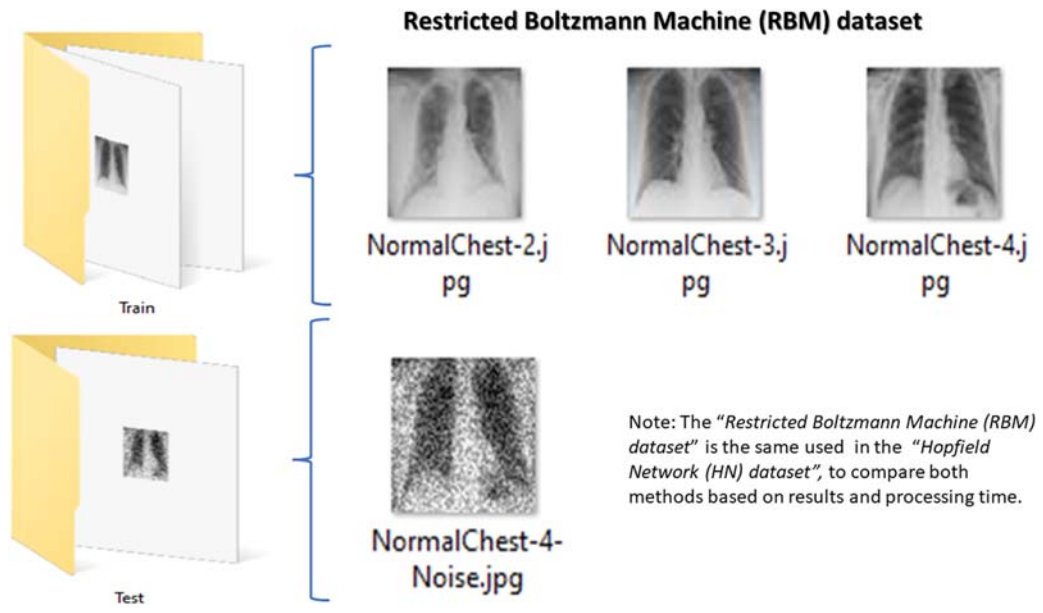
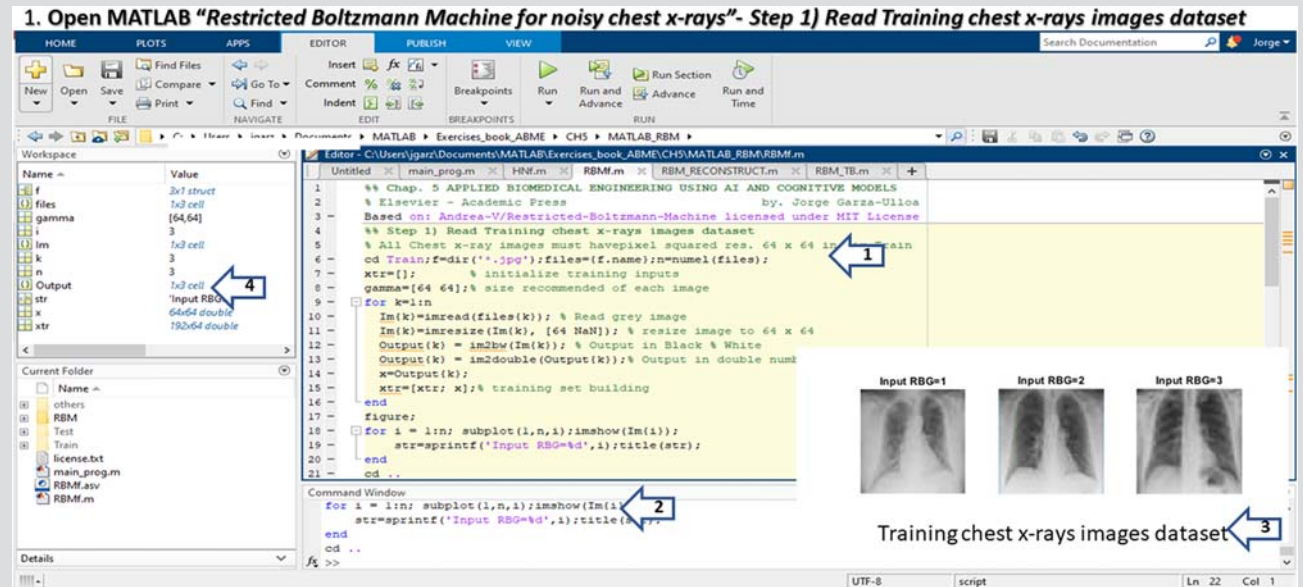


FIGURE 6.11 Restricted Boltzmann Machine (RBM) dataset.

Table of slides 6.4 Steps for MATLAB Deep Learning Toolbox to obtain RBM model from MATLAB Deep Learning Toolbox to reconstruct noisy chest X-ray images and compare performance and results obtained from the HN in Research 5.10.

Slide Description Screen figure

1 Open MATLAB “Restricted Boltzmann Machine for noisy chest X-rays”—Step 1) Read Training chest X-ray images dataset
Go to the directory “...\\Exercises_book_ABME\\CH6\\MATLAB_RBM.” Open the script “RBMf.m.” Copy and paste in the command prompt: “Step 1) Read Training chest X-ray images dataset.” Where the original images are read as 64×64 pixels RGB images, then converted to binary format and store in the cell variable “Output”



Go to the directory “...\\Exercises_book_ABME\\CH6\\MATLAB_RBM”. Open the script “RBMf.m”. Copy and paste in the command prompt: “Step 1) Read Training chest x-rays images dataset”. Where the original images are read as 64×64 pixels RGB images, then converted to binary format and store in the cell variable “Output”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

2 MATLAB “Restricted Boltzmann Machine for noisy chest X-rays” – Step 2) Train all Chest X-ray images stored in the train directory Copy and paste in the command prompt: “Step 2) Train all Chest X-ray images from directory Train.” For this example, the MATLAB user function “RBM_TB()” by Andrea-V/ Restricted-Boltzmann-Machine licensed under MIT License is used [77]. Therefore the net options are specified, and the RBM model obtained from the trained images is store in the MATLAB “net” variable as a structure

2. MATLAB “Restricted Boltzmann Machine noisy chest x-rays”- Step 2) Train all Chest x-ray images stored in the train directory

The screenshot displays the MATLAB IDE interface. The main editor window shows the following code:

```
23 %% Step 2) Train all Chest x-ray images from directory Train
24 %% Training Options
25 Options.max_itera=100; % maximum number of learning iterations
26 Options.N_gs=60; % number of gibbs sampling steps
27 Options.N_neurons=gamma(2); % number of neurons in the hidden layer
28 Options.eps=0.01; % learning rate
29 Options.Sz_sb=10; % size if mini-batch of data
30 %% Training process
31 disp('Training images for RBM.. wait');
32 net=RBM_TB(xtr,Options);% training
33 net.Tr_acc;
34 disp('Training images for HN... ready');
35 figure;
36 for i = 1:n; subplot(1,n,1);imshow(Output(i));
37 str=sprintf('Input B&W=%d',i);title(str);
38 end
```

The Command Window shows the following output:

```
net=RBM_TB(xtr,Options);% training
net.Tr_acc;
disp('Training images for HN... ready');
figure;
for i = 1:n; subplot(1,n,1);imshow(Output(i));
str=sprintf('Input B&W=%d',i);title(str);
end
Training images for RBM.. wait
Training images for HN... ready
```

The Workspace window shows the following variables:

Name	Value
ans	0.0983
f	2x1 struct
files	2x2 cell
gamma	[64,64]
i	3
lm	2x2 cell
k	3
n	3
net	1x1 struct
Options	1x1 struct
Output	2x2 cell
str	'Input B&W=3'

The Command Window also displays three chest X-ray images labeled "Input B&W=1", "Input B&W=2", and "Input B&W=3". Below these images, the text "Trained chest x-rays images by RBM" is displayed.

Copy and paste in the command prompt: “Step 2) Train all Chest x-ray images from directory Train”. For this example, the MATLAB user function “RBM_TB()” by Andrea-V/Restricted-Boltzmann-Machine licensed under MIT License. Therefore, the net options are specified, and the RBM model obtained from the trained images is store in the MATLAB “net” variable as a structure.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

3

MATLAB “Restricted Boltzmann Machine for noisy chest X-rays” – Step 3) Load a test image stored from the test directory
Copy and paste in the command prompt: “Step 3) Load test image from the directory Test.” Only the first image stored is read, and it is shown in a Figure as “Test Input RGB = 1” and its conversion to binary as “Test Input B&W = 1”

3. MATLAB “Restricted Boltzmann Machine for noisy chest x-rays”- Step 3) Load a test image stored from the test directory

The screenshot shows the MATLAB Editor interface. The workspace on the left contains variables like 'ans', 'f', 'files', 'gamma', 'i', 'Im', 'k', 'm', 'n', 'net', 'Options', and 'Output'. The Command Window at the bottom shows the execution of the script. The code in the Editor is as follows:

```
40 %% Step 3) Load test image from directory Test
41 % One Chest x-ray images with noise pixel squared res. 64 x 64 in dir Test
42 -
43 od Test;
44 - f=dir('*.jpg'); files=(f.name); m=1; % only one image to test
45 -
46 for k=1:m
47     Im(k)=imread(files(k));
48     Im(k)=imresize(Im(k), [64 NaN]); % resize image to 64 x 64
49     OutputT(k) = im2bw(Im(k)); %Output = im2bw(Output)
50     OutputT(k) = im2double(OutputT(k));
51 end
52 figure;
53 subplot(2,2,1); imshow(Im(1)); str=sprintf('Test Input R
54 subplot(2,2,2); imshow(OutputT(1)); str=sprintf('Test In
55 end
56 od ..
57 xp=OutputT(m);
```

Two figure windows are displayed: "Test Input RGB=1" showing a noisy grayscale chest X-ray image, and "Test Input B&W=1" showing the binary (black and white) version of the same image. Arrows labeled 1, 2, and 3 point to the code line for loading the image, the imshow call for the RGB image, and the overall figure window respectively.

Copy and paste in the command prompt: “Step 3) Load test image from the directory Test”. Only the first image stored is read, and it is shown in a figure as “Test Input RGB=1” and its conversion to binary as “Test Input B&W=1”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 4 Description

MATLAB "Restricted Boltzmann Machine for noisy chest X-rays" – Step 4) HM compares test with trained images & reconstruct it. Copy and paste in the command prompt: "Step 4) HM compares test with trained images and reconstruct it." Where the test image is fixed by the "RBM model," this was obtained from trained images at Step 2), and it shows the resulting "reconstructed chest X-ray image." Close all

Screen figure

4. MATLAB "Restricted Boltzmann Machine noisy chest x-rays"-Step 4) RBM compares test with trained images & reconstruct it

```
57 % Step 4) HM compares test with trained images and reconstruct it
58 % Ai=cell2mat(OutputT);
59 % outC=Ai(:); % convert matrix to vector
60 % xp=outC;
61 [y]=RBM_RECONSTRUCT(net,xp);
62 figure;
63 subplot(2,2,1);imshow(xp);str=sprintf('Test Input B&W=%d',1);title(str);
64 subplot(2,2,2);imshow(y);str=sprintf('Reconstructed =%d',1);title(str);
```

Command Window

```
>> % Step 4) HM compares test with trained images and reconstruct it
% Ai=cell2mat(OutputT);
% outC=Ai(:); % convert matrix to vector
% xp=outC;
[y]=RBM_RECONSTRUCT(net,xp);
figure;
subplot(2,2,1);imshow(xp);str=sprintf('Test Input B&W=%d',1);title(str);
subplot(2,2,2);imshow(y);str=sprintf('Reconstructed =%d',1);title(str);
>>
>>
>>
>>
>>
>>
>>
>>
>>
>>
```

Test Input B&W=1

Reconstructed =1

Reconstructed Noisy chest x-rays image by RBM

Copy and paste in the command prompt: "Step 4) HM compares test with trained images and reconstruct it". Where the test image is fixed by the "RBM model", this was obtained from trained images at step 2, and it shows the resulting "reconstructed chest x-ray image". Close all.

Conclusions

"RBM model obtained from MATLAB Deep Learning Toolbox allows the reconstruction of noisy chest X-ray images, comparing the test image with the trained images. The RBM model is faster but less precise than the HN model results obtained in Research 6.3.

Recommendation

This kind of "RBM models" can be used to store "images paths," that can be used in many biomedical engineering applications to identify many diseases and injuries in a short time.

6.2.9 Liquid State Machine

"Liquid State Machine (LSM)" is a special type identified as a "reservoir network," where the word "Liquid" comes from the analogy drawn to dropping a stone in water or other liquid to generate ripples in it. "LSM" was developed from the "computational neuroscience neural model" with real-time computations, transforming the time-varying inputs stream to a higher dimensional space. Basically, "LSM" is biological inspired in the "neural and synaptic states of the human neurons" incorporating the time as a variable and can be represented in four stages, as shown in Fig. 6.12A: "Input streams," "Liquid neurons or microcircuits," "State," and "readout neurons," where:

- "Input streams" is a time series data stream, in this step of the "LSM" the mapping of input streams to output streams is executed.

- "Liquid neurons or microcircuits" is a "recurrent neural network of spiking neurons" that simulate how "brain neurons that are connected from axon terminals to dendrites to the next neurons" in a process known as "type I synapse," as explained in Section 1.4.10. Also, this stage acts as a preprocessor (temporal). This stage consists of a large collection of "nodes or neurons" configured as a "Spiking Neural Network (SNN)," these are "recurrent artificial neural networks that more closely mimic the brain-inspired neuromorphic computing that operates in the temporal domain" and their computation is based on the time resources available. Each node receives "time-varying input from the input streams" and from other nodes. The nodes in the "microcircuits" are randomly connected to each other, and the recurrent nature of the connections turns the time-varying input into a spatio-temporal pattern of activations in the network nodes. The spatiotemporal patterns of activation are read out by linear discriminant units.
- "State" is the stage to measure the state of the liquid at any given time (t).
- "Readout neurons" are the "synaptic plasticity values" that represent the strength or weaken over time, with the assumption that they have no temporal integration capability of their own.

"LSM" equations are related with "neurons type I synapse" as indicated in Fig. 6.12B. The equations of "neural and synaptic states of the human neurons" on time (t) for

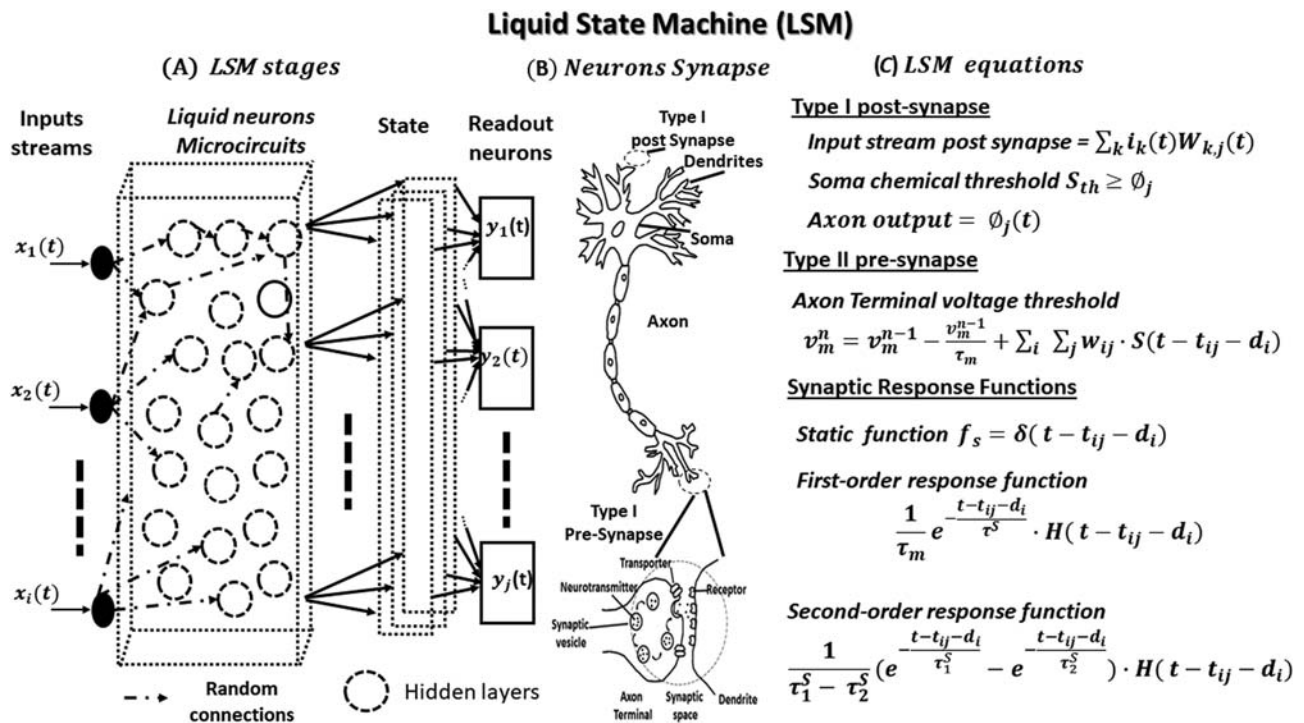


FIGURE 6.12 Liquid State Machine (LSM): (A) LSM stages, (B) Neuron synapse, and (C) LSM equations.

an “LSM” can be divided into three groups: “*type 1 postsynapse*,” “*type 1 presynapse*,” and “*synaptic response functions*,” as summarized in Fig. 6.12C, where:

- “*Type 1 postsynapse*” occurs between the axon terminal from the last neuron and the dendrites from the actual neuron, and can be evaluated as “*Input stream from the postsynapse*,” “*Soma chemical threshold*,” and “*axon output*.”
- “*Type 1 presynapse*” occurs between the axon terminal from the actual neuron and the dendrites of the next neuron, and can be evaluated as a “*axon terminal voltage threshold*.”
- “*Synaptic response functions*” consist of the adding of three components, and can be evaluated as “*static function*,” “*first-order response function*,” and “*second-order function*.”

There is not a standard definition for the generation of neural networks; researchers up to now have visualized three of them as:

- “*First Generation NN*” when they are based on “*perceptron or threshold gates*” and the direction of flow is “*feed forward*.” Here the output is binary, and it could be multilayered as well and any Boolean function can be implemented by “*MLP (Multilayer perceptron)*.”
- “*Second Generation NN*” when they are based on data flow in both directions with continuous output values.
- “*Third Generation NN*” when they are based on a “*Spiking Neural Network*” that uses discrete events in time rather than continuous values. Where a spike is determined resolving differential equations, and they are fired when a potential or threshold value is crossed, in this way resemble a more biologically realistic model of neurons firing, getting closer to the gap between “*Neuroscience*” that explained how the brain neurons are fired applying “*Deep Learning models based on artificial neural network*.”

Some “LSM” examples used in “*Biomedical Engineering*” are:

- “*Anytime multipurpose emotion recognition from EEG data using a Liquid State Machine based framework*” by Zoubi et al. [22]. In this a “*Liquid State Machines (LSM)*” is used to recognize the emotional state of an individual based on “*EEG*” data. The “*LSM*” model is applied for automatic feature extraction and prediction from raw “*EEG*” with potential extension to a wider range of applications including a multipurpose and anytime recognition framework, used to train a model to predict valence, arousal, and liking levels at different durations of the input.
- “*Performance and robustness of bio-inspired digital liquid state machines: A case study of speech*

recognition” by Jin and Li [23]. This paper presents a systematic performance and robustness study of “*bio-inspired digital liquid state machines (LSMs)*” for the purpose of future hardware implementation because of their low overhead, good robustness, and high recognition performance.

- “*Surrogate-Assisted Evolutionary Search of Spiking Neural Architectures in Liquid State Machines*” by Zhou et al. [24]. As a class of recurrent Spiking Neural Network (SNN), “*Liquid State Machines (LSMs)*” are biologically more plausible models imitating the architecture and functions of the human brain for information processing. However, few LSM models can outperform conventional analog neural networks for solving real-world classification or regression problems, which can mainly be attributed to the sensitivity of the training performance to the architecture of the reservoir and the parameters in the spiking neuron models. This research uses a surrogate-assisted evolutionary search method for optimization of the hyperparameters and neural architecture of the reservoir of “*LSMs*” using the “*covariance matrix adaptation evolution strategy (CMA-ES)*.” For reducing the search space, the architecture of the “*LSM*” is encoded by a connectivity probability together with the hyperparameters in the spiking neuron models. To enhance the computational efficiency, a Gaussian process is adopted as the surrogate to assist the “*CMA-ES*.”

Please see the next example: Research 6.5, section 6.2.10.1 that creates a Reservoir Computing approach for a simulation of “*Liquid State Machine (LSM)*” of node-neurons from “*Spiking Neural Networks (SNN)*” based on the “*Izhikevich neuronal mathematical model*” to differentiate normal and pneumonia on chest X-rays.

In summary, like other kinds of neural networks, “LSM” is based around the neurobiology of the human brain but includes the variable of “time.” In “LSM” the “input streams” are represented as the motion of the falling stone in a liquid, which has been converted into a “spatiotemporal pattern” of liquid displacement representing the ripples, then the ripples can be evaluated to understand what is happening in the system being studied [10]. “LSM” has many biomedical applications in speech and audio recognition, image pattern recognition, music classification, robot path planning, fingerprint scanners, facial emotion recognition, and others [25].

6.2.10 Echo State Network

“*Echo State Network (ESN)*” is a special type identified as a “*reservoir network*” like “*LSM*.” “*ESN*” has memory properties by way of the feedback loops as a “*recurrent*

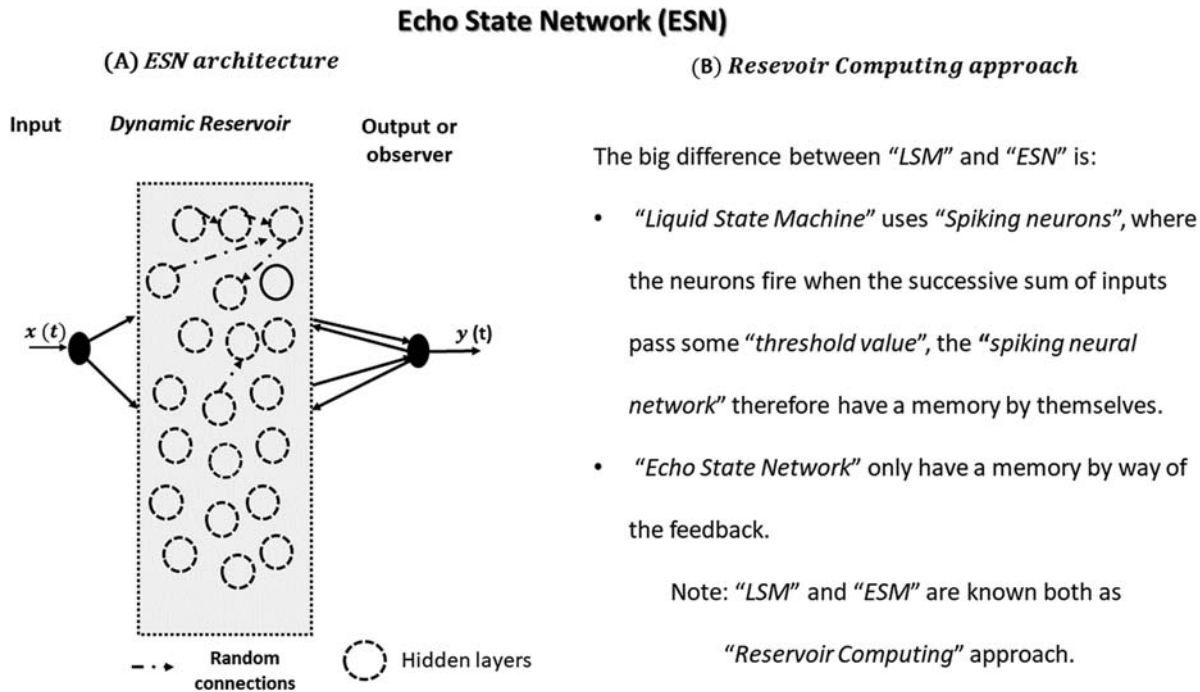


FIGURE 6.13 Echo State Network (ESN): (A) ESN architecture and (B) Reservoir Computing approach.

neural network (RNN)” using supervised learning. An “*ESN*” is not organized in a standard set of layers, its architecture is shown in Fig. 6.13A. It can be represented with: “input,” “a dynamic reservoir,” and “output or observer.” The “dynamic reservoir” has random connections between the neurons, and the training is as follows: instead of feeding the input and backpropagating the error, the input is fed, it is forwarded, and it updates the neurons for a while and observes the output or observer over time. During training, only the connections between the “output (observer)” and the hidden units in the “dynamic reservoir” are changed [26]. “*ESN*” tends to use neurons with a “sigmoid activation function.” The output of its neurons at a given time is therefore a function of the input to the neurons in the previous time step only.

The big difference between “*LSM*” and “*ESN*” as indicated in Fig. 6.13B, is that “*Liquid State Machine*” uses “*Spiking neural network*,” where the neurons fire when the successive sum of inputs pass some “*threshold value*”; the “*spiking neurons*” therefore have a memory by themselves, while in the “*ESM*” they only have a memory by way of the feedback. “*LSM*” and “*ESM*” are both known as “*reservoir computing*” approaches.

Some “*ESM*” examples used in “*Biomedical Engineering*” are:

- “*Decoding electroencephalographic signals for direction in brain–computer interface using echo state*

network and Gaussian readouts” by Kim and Jeong [27]. Noninvasive “*brain–computer interfaces (BCI)*” for body movement control via an electroencephalogram (EEG) have been extensively investigated; in this research a novel direct decoding method is described for user intention about the movement directions using the “*echo state network and Gaussian readouts*.” Importantly parameters in the network were optimized using the “*genetic algorithm*” method to achieve better decoding performance of up to 95% accuracy.

- “*Multiobjective ensembles of echo state networks and extreme learning machines for streamflow series forecasting*” by Ribeiro et al. [28]. In this research the development of ensembles of unorganized machines, namely “*Extreme Learning Machines (ELMs)*” and “*Echo State Networks (ESNs)*,” are used for two primary contributions: (1) a new training logic for “*ESNs*” that enables the application of bootstrap aggregation (bagging); and (2) the employment of multiobjective optimization to select and adjust the weights of the ensemble’s base models, taking into account the trade-off between bias and variance.
- “*Fault diagnosis model based on Granular Computing and Echo State Network*” by Lu et al. [29]. In this paper, to improve the efficiency and accuracy of electronic equipment fault diagnosis, a fault diagnosis model based on “*Granular Computing*” and “*Echo State Network (ESN)*” is proposed. Firstly, the attribute reduction of the test index is carried out based on a “*granular computing*

model.” An attribute distinguishing ability index is defined based on attribute value influence degree, and on the stage of fault identification by an “ESN.”

6.2.10.1 Research 6.5 Create a Reservoir Computing approach for a simulation of “Liquid State Machine (LSM)” of node-neurons from “Spiking Neural Networks (SNN)” based on the “Izhikevich neuronal mathematical model” to differentiate normal and pneumonia on chest X-rays

6.2.10.1.1 Case for research

“Obtain an LSM model from MATLAB to create a Reservoir Computing approach for a simulation of “Liquid State Machine (LSM)” of node-neurons from “Spiking Neural Networks (SNN)” based on the “Izhikevich neuronal mathematical model” to “differentiate normal and pneumonia on chest X-rays.”

6.2.10.1.2 General objective

Apply “MATLAB” to define, build, and make “a simulation of “Liquid State Machine (LSM)” of node-neurons from “Spiking Neural Networks (SNN),” based on the “Izhikevich neuronal mathematical model” to “differentiate normal and pneumonia on chest X-rays.”

6.2.10.1.3 Specific objectives

- Define a “Liquid State Machine (LSM)” to analyze images using black pixels as a matrix and a vector for readout neurons.

- Read input images as shown at Fig. 6.14, and define “SNN net parameters”.
- Initialize (SNN): Receptors variables for simulation.
- Simulation LSM: input stream in time, microcircuit, state, and readout neurons.
- Display LSM: State and readout minimum neurons.

6.2.10.1.4 Background for “Izhikevich neuronal mathematical model”

Izhikevich neuronal mathematical [30] model reproduces “spiking (shown in Fig. 6.15G) and bursting behavior of known types of cortical neurons.” This model combines the “Hodgkin–Huxley-type dynamics and the computational efficiency of integrate-and-fire neurons.” The “Hodgkin–Huxley-type neuronal models” uses a two-dimensional (2D) system of ordinary differential equations, as shown in Fig. 6.15H, and eq. 1) and eq. 2) are obtained by fitting of the spike initiation dynamics of a cortical neuron in millivolts, and eq. 3) as the reset value condition. The resting potential is between -70 and -60 mV, defined by the value of “ b .” In real neurons the threshold can be from -55 to -40 mV. The dimensionless variables and dimensionless parameters meanings are:

- “ v ” represents the membrane potential of the neuron.
- “ u ” represents the membrane recovery.
- “ a ” describes the timescale of the recovery variable “ v .” Its typical value is “ $a = 0.02$.”
- “ b ” describes the sensibility of the recovery variable “ u .” Its typical value is “ $b = 0.2$.”
- “ c ” describes the after-spike reset value of the variable “ v .” Its typical value is “ $b = -65$ mV.”

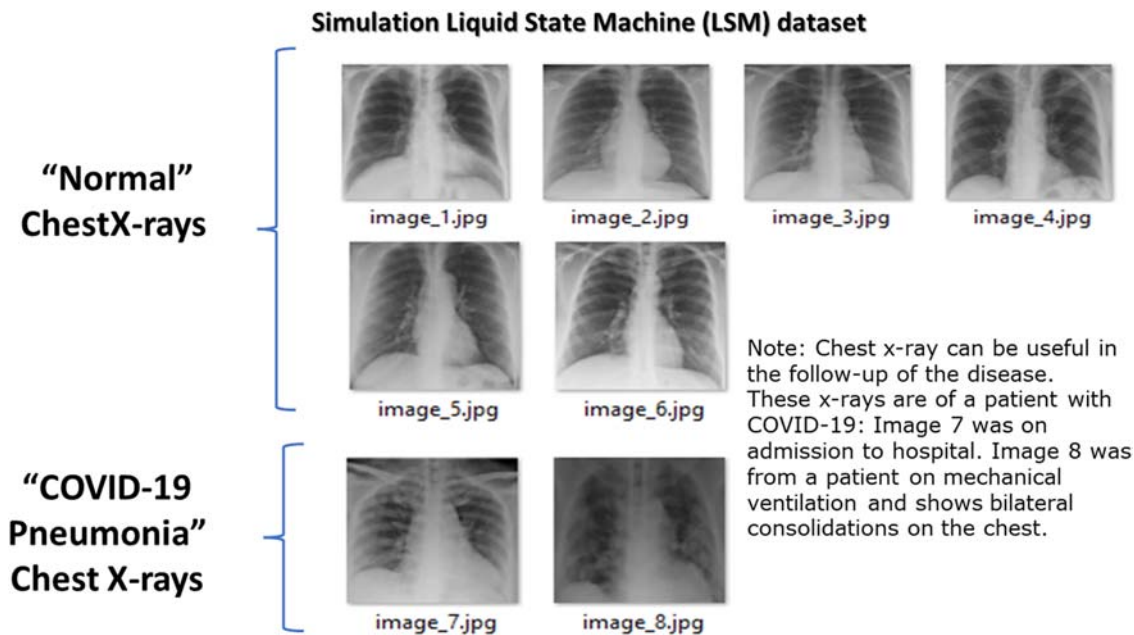


FIGURE 6.14 Simulation Liquid State Machine (LSM) dataset.

Simulation Liquid State Machine (LSM)

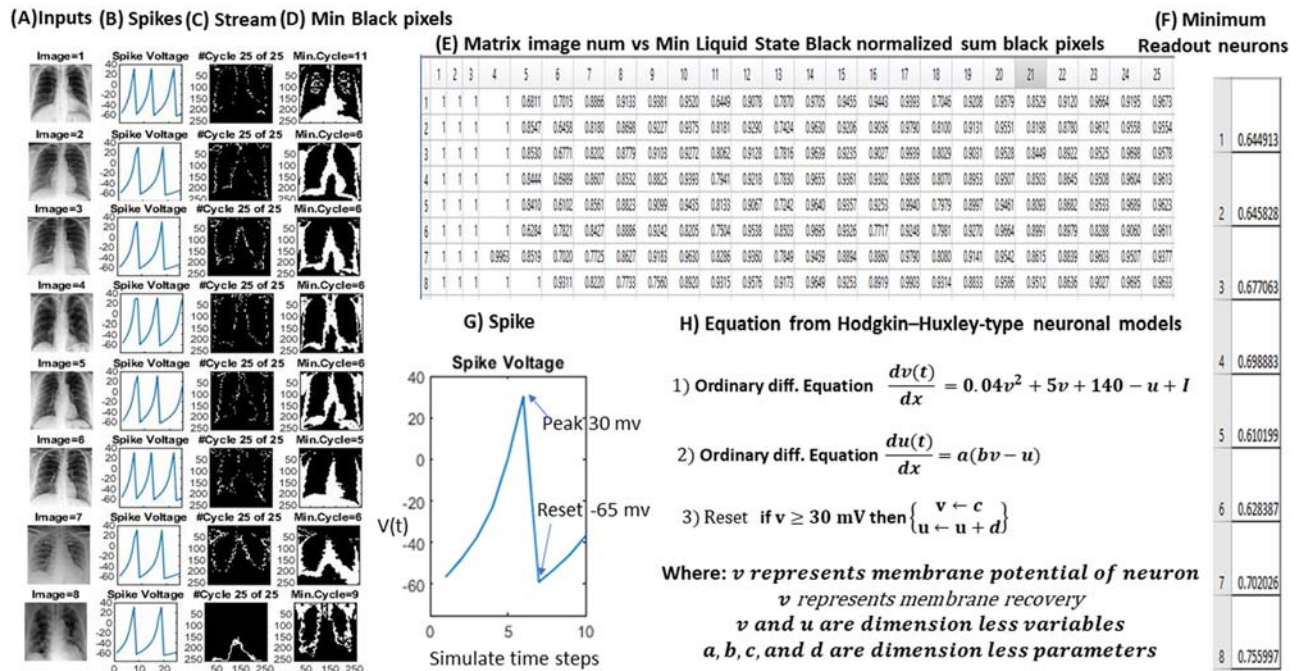


FIGURE 6.15 Simulation Liquid State Machine (LSM) obtained in Research 6.5: (A) input images, (B) neuron spikes, (C) input stream on time, (D) min black pixels of cycle, (E) matrix image num versus min liquid state black normalized sum black pixels, (F) minimum of all readout neurons, (G) spike voltage of neuron, and (H) equation from Hodgkin-Huxley-type neuronal models.

- “ d ” describes the after-spike reset value of the variable “ u ” caused by slow high-threshold of Na^+ and K^+ conductances. Its typical value is “ $d = 2$.”

6.2.10.1.5 Background for “COVID-19 lung imaging chest X-rays”

“Chest X-rays” is one of the tests that may be ordered to evaluate “pneumonia” others are: “CT of the lungs,” which shows finer detail that is hard to see on plain chest X-rays; “ultrasound of the chest” to detect fluid surrounding the lungs; “MRI of the chest” that provides additional information about the cause or extent of the abnormalities; and “needle biopsy of the lung” that helps to determine the cause of the “pneumonia.” A “chest X-ray exam” allows doctors to see lungs, heart, and blood vessels to help determine if a patient has pneumonia. When interpreting the chest X-ray, the radiologist will look for “white spots in the lungs (called infiltrates) that identify an infection.” This exam will also help to determine any complications related to pneumonia, such as abscesses or pleural effusions (fluid surrounding the lungs) [31].

“COVID-19 lung imaging” is generally indicated in any COVID-19 patient with worsening respiratory

status [32]. Some of the more common abnormalities seen on “chest X-rays (CXR)” are:

- “Ground-glass opacities distributed bilaterally in bases and peripheries distributed bilaterally in bases and peripheries.” On admission, ground-glass opacity was the most common radiologic finding on chest “Computed Tomography (CT),” and later can be visualized on “CXR” [33].
- “Chest films can be useful in the follow-up of the disease.” It evolves rapidly and lung involvement is associated with severity; findings progress over 1–3 weeks, typically peaking at 10–14 days, findings progress with time and appear worse at day 10–12 [34].
- “Chest images findings may be present in asymptomatic individuals or presymptomatic individuals and may be absent early in the course of disease.”
- In general, the “lung imaging findings of COVID-19 patients” are consistent with other viral pneumonias—there is no proven specific finding for COVID-19, though there may be suggestive patterns. Some features found are:
 - Bilateral shadowing (72.9%)—mostly ground-glass opacity (68.5%),
 - Unilateral disease (25%),

- Local patchy shadowing,
- Interstitial abnormalities (less common finding, <5% in some studies),
- Pleural effusions are uncommon.

6.2.10.1.6 Dataset

The dataset consists of “*eight CXR images*”: 1 to 6 are from “*Normal Chest X-ray*” patients, and 7 and 8 are from “*COVID-19 lung imaging*” patients. Image 7 was on admission to hospital. Image 8 was from a patient on mechanical ventilation and shows bilateral consolidations on the chest.

Note: This images dataset is available in the companion directory of the book, in the following folder “`..\Exercises_book_ABME\CH6\MATLAB_LSM`.”

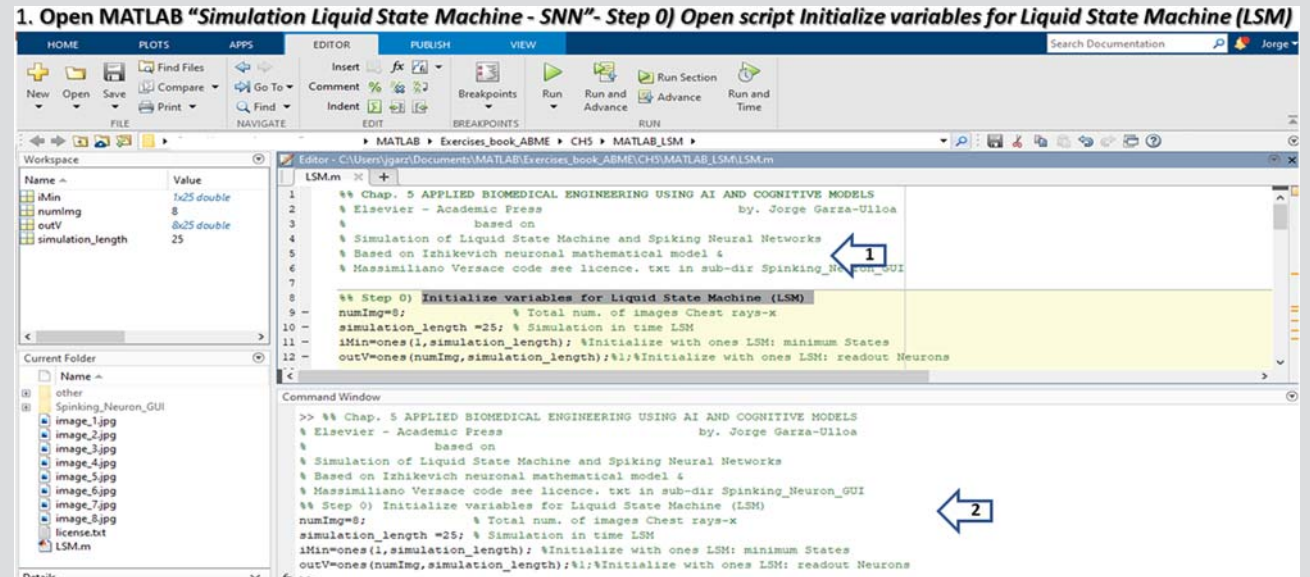
6.2.10.1.7 Procedure

The steps to “*Obtain an LSM model from MATLAB to create a Reservoir Computing approach for a simulation of “Liquid State Machine (LSM)” of node-neurons from “Spiking Neural Networks (SNN)”*” are summarized in *Table of slides 6.5*, and each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

Table of slides 6.5 Steps for MATLAB to “Obtain an AI model from MATLAB to create a Reservoir Computing approach for a simulation of “Liquid State Machine (LSM)” of node-neurons from “Spiking Neural Networks (SNN).”

Slide Description Screen figure

1 Open MATLAB “Simulation Liquid State Machine —SNN—Step 0) Open script Initialize variables for Liquid State Machine (LSM) Go to the directory “... \Exercises_book_ABME\CH6\MATLAB_LSM.” Open the script “LSM.m.” Copy and paste in the command prompt: “Step 0) Initialize variables for Liquid State Machine (LSM) “numImages” is the number of images to be analyze, “simulation_length” is the number of states when an images is analyzed, “iMin” is a matrix with one where the minimum of normalized black pixels will be stored, and “outV” is a vector to store the “LSM: readout neurons”



Go to the directory “... \Exercises_book_ABME\CH6\MATLAB_LSM”. Open the script “LSM.m”. Copy and paste in the command prompt: “Step 0) Initialize variables for Liquid State Machine (LSM)”. “numImages” is the number of images to be analyze, “simulation_length” is the number of states when an images is analyzed, “iMin” is a matrix with one where the minimum of normalized black pixels will be stored, and “outV” is a vector to store the “LSM: readout neurons”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

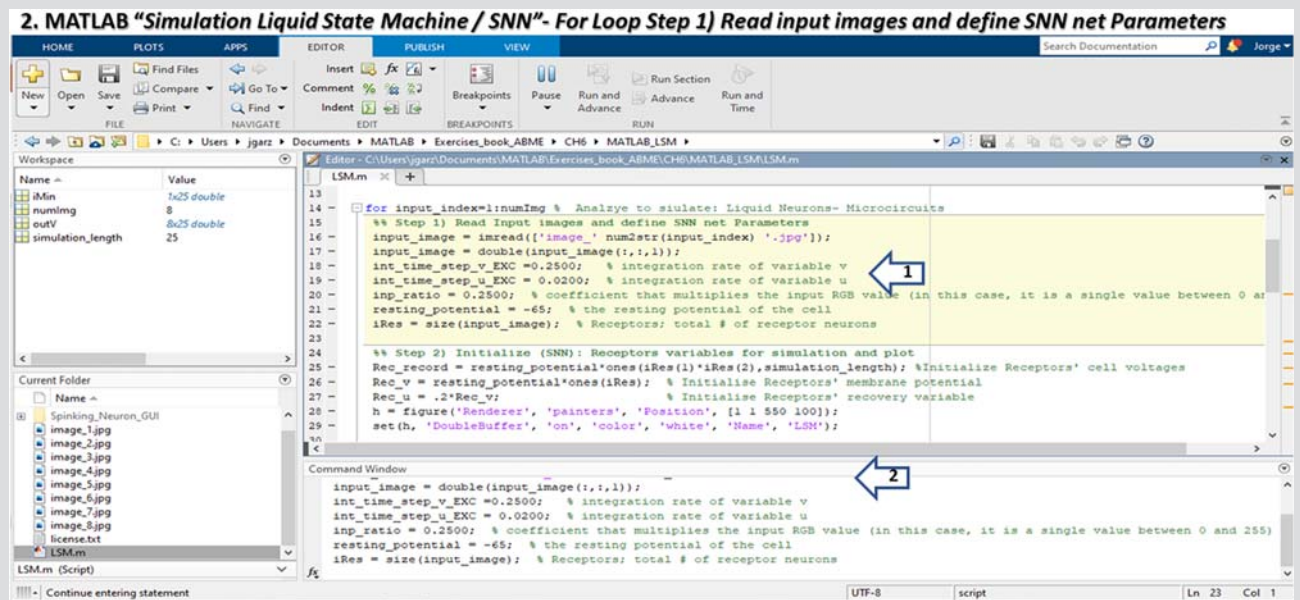
Slide 2 Description

MATLAB “Simulation Liquid State Machine/SNN”—For Loop Step 1) Read input images and define SNN net Parameters

Copy and paste in the command prompt: “For Loop Step 1) Read input images and define SNN net Parameters.” In each loop an image is analyzed by the “SNN” to generate the “Liquid neurons/microcircuit,” plot the results and store the numeric value of “state” represented as the normalized min. black pixel

Note: These instructions are inside a loop, that is closed on Step 2)

Screen figure



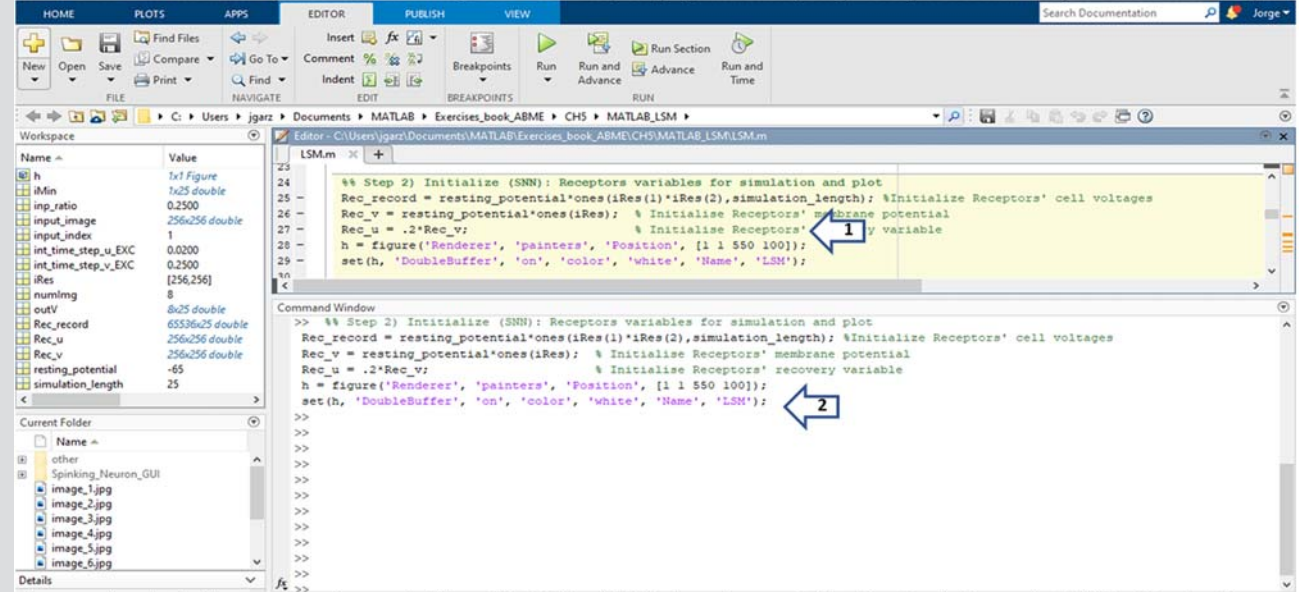
Copy and paste in the command prompt: “For Loop Step 1) Read input images and define SNN net Parameters”. In each loop an image is analyzed by the “SMM” to generate the “Liquid neurons/microcircuit”, plot the results and store the numeric value of “state” represented as the normalized min. black pixel. Note: These instructions are inside a for-loop, that is closed on step 2)

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

3

MATLAB “Simulation Liquid State Machine/SNN”—For Loop Step 2) Initialize (SNN): Receptors variables for simulation and plot
Copy and paste in the command prompt: “2) Initialize (SNN): Receptors variables for simulation and plot.” In this step the variable needed to apply the “Izhikevich neuronal mathematical model” are specified as shown in Fig. 6.15H

3. MATLAB “Simulation Liquid State Machine/SNN”—For Loop Step 2) Initialize (SNN): Receptors variables for simulation and plot



Copy and paste in the command prompt: “Step 2) Initialize (SNN): Receptors variables for simulation and plot”. In this step the variable needed to apply the “Izhikevich neuronal mathematical model” are specified as shown in figure 6.15H.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

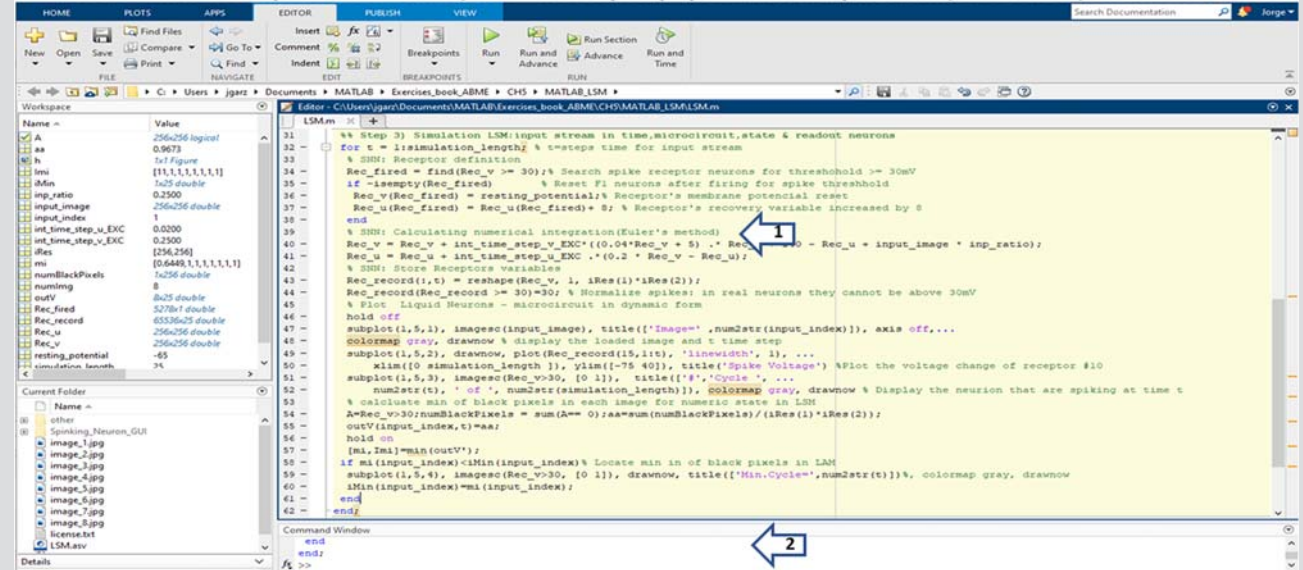
(Continued)

Slide Description

Screen figure

4 MATLAB “Simulation Liquid State Machine/SNN”—For Loop Step 3) Simulation: input stream, microcircuit, state & readout neurons
Copy and paste in the command prompt: “Step 3) Simulation LSM: input stream in time, microcircuit, state & readout neurons.” Here the “SNN: receptor definition,” the search for values of “voltage >30 mV” is made to reset the membrane, then the “Euler integration” is calculated, the plots are generated for each step, and the matrix “outV” is generated

4. MATLAB “Simulation Liquid State Machine/SNN”- For Loop Step 3) Simulation: input stream, microcircuit, state & readout neu.



```
LSM.m
% Step 3) Simulation LSM: input stream in time, microcircuit, state & readout neurons
31 for t = 1:simulation_length; % t=steps time for input stream
32 % SNN: Receptor definition
33 Rec_fired = find(Rec_v >= 30); % Search spike receptor neurons for threshold >= 30mV
34 if ~isempty(Rec_fired) % Reset FI neurons after firing for spike threshold
35 Rec_v(Rec_fired) = resting_potential; % Receptor's membrane potential reset
36 Rec_u(Rec_fired) = Rec_u(Rec_fired) + S; % Receptor's recovery variable increased by S
37 end
38 % SNN: Calculating numerical integration(Euler's method)
39 Rec_v = Rec_v + int_time_step_v_EXC*(0.04*Rec_v + 5) .* Rec_u - Rec_u + input_image * inp_ratio;
40 Rec_u = Rec_u + int_time_step_v_EXC.*(0.2 * Rec_v - Rec_u);
41 % SNN: Store Receptors variables
42 Rec_record(Rec_record >= 30) = reshape(Rec_v, 1, iRes(1)*iRes(2));
43 % Plot Liquid Neurons - microcircuit in dynamic form
44 hold off
45 subplot(1,5,1), imagesc(input_image), title(['Image', num2str(input_index)]), axis off,...
46 colormap gray, drawnow % display the loaded image and t time step
47 subplot(1,5,2), drawnow, plot(Rec_record(15,1:t), 'linewidth', 1), ...
48 xlim([0 simulation_length]), ylim([-75 40]), title('Spike Voltage') %Plot the voltage change of receptor #10
49 subplot(1,5,3), imagesc(Rec_v>30, [0 1]), title(['t', 'Cycle', ...
50 num2str(t), ' of ', num2str(simulation_length)]), colormap gray, drawnow % Display the neuron that are spiking at time t
51 % calculate min of black pixels in each image for numeric state in LSM
52 A=Rec_v>30;numBlackPixels = sum(A== 0);aa=sum(numBlackPixels)/(iRes(1)*iRes(2));
53 hold on
54 [mi, Imi]=min(outV');
55 if mi (input_index)<imin(input_index) % Locate min in of black pixels in LSM
56 subplot(1,5,4), imagesc(Rec_v>30, [0 1]), drawnow, title(['Min.Cycle', num2str(t)]), colormap gray, drawnow
57 %Min(input_index)=mi(input_index);
58 end
59 end
60 end
61 end
62 end
Command Window
end
end;
fx >>
```

Copy and paste in the command prompt: “Step 3) Simulation LSM: input stream in time, microcircuit, state & readout neurons”. Here the “SNN: receptor definition,” the search for values of “voltage > 30 mV” is made to reset of the membrane, then the “Euler integration” is calculated, the plots are generated for each step, and the matrix “outV” is generated.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

5

MATLAB “Simulation Liquid State Machine/SNN”—For Loop Step 4) Display LSM: State and Readout minimum Neurons.

Copy and paste in the command prompt: “Step 4) Display LSM: State & Readout minimum Neurons.”

Here the “LSM: Liquid neurons/microcircuits” as shown in Fig. 6.15E and the “LSM: readout minimum neurons results” are indicated in Fig. 6.15F

5. MATLAB “Simulation Liquid State Machine/SNN”—For Loop Step 4) Display LSM: State & Readout minimum Neurons

The screenshot displays the MATLAB environment with the following components:

- Workspace:** Lists variables such as A (256x256 logical), aa (0.9633), h (1x1 Figure), lmi ([11,6,6,6,5,6,9]), lMin (1x25 double), inp_ratio (0.2500), input_image (256x256 double), input_index (8), int_time_step_u_EXC (0.0200), int_time_step_v_EXC (0.2500), iRes (256,256), mi ([0.6449, 0.6458, 0.6771, 0.6989, 0.6102, 0.6284, 0.7020, 0.7560]), numBlackPixels (7x256 double), numimg (8), and outV (8x25 double).
- Editor - LSM.m:** Contains code for Step 4. Line 64 is a comment. Line 65 is another comment. Line 66 is a comment. Line 67: `disp('LSM - State in outV variable see figure 5.41 d')`. Line 68: `% mi is a vector normalized variable with min of all steps min black pixels`. Line 69: `disp('LSM- Readout minimum neurons see figure 5.41 f')`. Line 70: `mi`. Line 71: `mi =`.
- Command Window:** Shows the execution of the code. The output for line 67 is: `LSM - State in outV variable see figure 5.41 d`. The output for line 69 is: `LSM- Readout minimum neurons see figure 5.41 f`. Below this, the variable `mi` is displayed as a row vector: `mi =`
`0.6449 0.6458 0.6771 0.6989 0.6102 0.6284 0.7020 0.7560`.

Copy and paste in the command prompt: “Step 4) Display LSM: State & Readout minimum Neurons”. Here the “LSM: Liquid neurons/microcircuits” as shown in figure 6.15 D” and the “LSM: readout minimum neurons results” are indicated in figure 6.15 f.

Conclusions

The AI model obtained created using a “Reservoir Computing approach” by a simulation an “Liquid State Machine (LSM)” of node-neurons using “Spiking Neural Networks (SNN),” based on “Izhikevich neuronal mathematical model” was apply in this research. This model allows the analysis of the “chest X-ray images” shown in Fig. 6.15A, where spiking voltage (as shown in Fig. 6.15B) fired the SNN at different steps (as shown in Fig. 6.15C) to locate the image with the minimum amount of normalized black pixels (as shown in Fig. 6.15D) Min Black pixels). With the purpose to find when the “COVID-19 lung imaging shows the maximum pneumonia damage”, all the calculation are stored on a Reservoir Computing (shown in Fig. 6.15E), represented as a Matrix of image num versus Min Liquid State Black normalized sum black pixels to obtain as the result the “minimum readout fired neurons” (as indicated in Fig. 6.15F) as a demonstrative example of “simulation of LSM” applying a “Recurrent Neural Network” type.

Recommendation

The “Reservoir Computing approach” can be used in many biomedical engineering applications where results are needed faster; it has been applied for images analysis, NLP, resolve complex biomedical equations, etc.

6.2.11 Korhonen Network

“Korhonen Network (KN) also known as Self-Organizing Map (SOM)” is a type of “artificial neural network (ANN)” used as unsupervised learning for classification. Input is presented to the network, after which the network assesses which of its neurons most closely match that input. They use a neighborhood function to preserve the topological properties of the input space. These neurons are then adjusted to match the input even better, dragging along their neighbors in the process. How much the neighbors are moved depends on the distance of the neighbors to the best matching units [35]. It seems to be the most natural way of learning, which is used in our brains, where no patterns are defined.

Some “Korhonen Network or Self-Organizing Map (SOM)” examples used in “Biomedical Engineering” are:

- “Analysis of Spatial Spread Relationships of Coronavirus (COVID-19) Pandemic in the World using Self-Organizing Maps” by Melinet al. [36]. This paper makes an analysis of the spatial evolution of coronavirus pandemic around the world by using “self-organizing maps (SOM).” Based on the clustering abilities of self-organizing maps we are able to spatially group together countries that are similar according to their coronavirus cases, in this way being

able to analyze which countries are behaving similarly and thus can benefit by using similar strategies in dealing with the spread of the virus, with the purpose of proposing successful strategies for similar countries.

- “Coupling bootstrap with synergy self-organizing map-based orthogonal partial least squares discriminant analysis: Stable metabolic biomarker selection for inherited metabolic diseases” by Yang et al. [37]. The “Kohonen self-organizing map” is a well-established variable reduction algorithm in identifying significant biomarkers based on variable clustering. In this paper a novel screening system is presented by “coupling bootstrap” with synergy “self-organizing map” based “orthogonal partial least squares discriminant analysis” for stable and biologically meaningful metabolic biomarker selection.
- “Assessing relationships between chromatin interactions and regulatory genomic activities using the self-organizing map” by Kunz et al. [38]. Few existing methods enable the visualization of relationships between regulatory genomic activities and genome organization as captured by Hi-C experimental data. In this paper an approach to the visualization and analysis of “chromatin organization based on the Self-Organizing Map (SOM)” for genome-wide Hi-C datasets is introduced. The “SOM” algorithm provides a 2D manifold which adapts to represent the high-dimensional chromatin interaction space. The resulting data structure can then be used to assess relationships between regulatory genomic activities and chromatin interactions.

As an example of “Korhonen Network (KN) or Self-Organizing Map (SOM),” see Research 5.2, section 5.3.2 MATLAB Deep Learning Toolbox using a Neural Network for clustering based on “Self-Organizing MAP through a Shallow Neural Network” to “analyze Surface Electromyography signals in Diabetes Type II patient,” where two matrices were analyzed:

1. Matrix based on 2 vectors normalize sEMG signals from “Type II DM patient right limb Soleus muscle” in two consecutive monthly visits.” “No correlation between the two signals were found, indicated an advance on the Soleus muscle affection response by the Type II DM.”
2. A matrix of 2 vectors inputs from two consecutive datasets of “healthy patient right limb Soleus muscle” obtained by “sEMG signals” on two monthly consecutive visits” shows a “positive correlation between them, indicating no significant changes on the behavior of the healthy Soleus muscle.”

6.3 Memory augmented neural networks types

“Memory augmented neural networks (MANN) or memory networks” imply the use of memory blocks hooked up to a neural network, allowing it to learn to reason. Some examples are shown in Fig. 1.11: *Neural Turing machines (NTM)*, *Differentiable Neural Computers (DNC)*, and others.

6.3.1 Neural Turing machine

The “*Neural Turing Machine (NTM)*” is the first application of “*Deep Learning (DL)*” to extend the capabilities of an “*Artificial Neural Network (ANN)*” by coupling them to external memory; it is based on a “*Recurrent Neural Network (RNN)*” coupled with external memory resources with which they can interact by attentional processes. The combined system is analogous to a “*Turing Machine or Von Neumann architecture*” but is differentiable end-to-end, allowing it to be efficiently trained with “*gradient descent*” that uses a feedback loop to adjust the model based on the error it observes between its predicted output and the actual output. Results demonstrate that “*NTM*” can infer simple algorithms such as copying, sorting, and associative recall from input and output examples.

“*NTM*,” like “*Von Neumann machine architecture*,” has the same three components, as shown in Fig. 6.16A:

“*controller*” that has two units—“*control unit and arithmetic/logic unit*” and “*memory unit*”; plus a place to store and retrieve data known as the “*Memory Unit*”; and the “*read and write heads*.” The “*memory*” is a “ $N \times W$ matrix,” where each row is called a “*location*.” The matrix will have “ N locations, with W -dimensional vectors.” And the “*attention mechanism*” defines some distributions called “*weightings*” over the “ N locations.” The units in which the “*weightings*” are produced and used are known as “*heads*,” and there are two types of them: “*read weighting*” for the read process as shown in equations in Fig. 6.16D 1), and “*write weighting*” for the writing process as shown in equations in Fig. 6.16D 2). It works basically as an algorithm that can store and retrieve information from a “*memory unit*” and process the information on the “*controller*.” The “*NTM controller*” is basically a “*Feed Forward Network (FFN) or Recurrent Neural Network (RNN)*” and the “*memory unit is a numeric matrix*” as shown in Fig. 6.16B.

Basically, to address the memory there are two types: “*Content-based addressing*” and “*Location-based addressing*,” where:

- “*Content-based addressing*” is when values selected from memory are based on the similarity of a comparison of a value in two steps, as shown in Fig. 6.16D 3). To accomplish this a “*Softmax layer*” can be used in

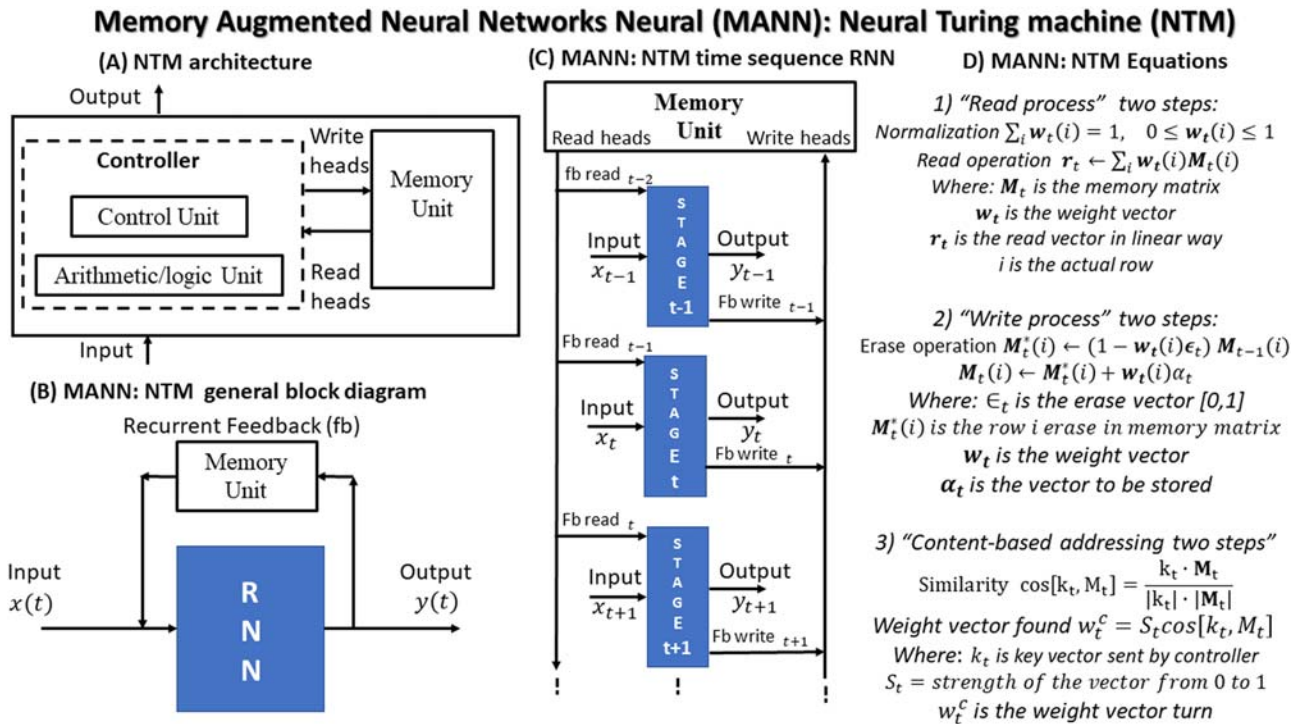


FIGURE 6.16 Memory Augmented Neural Networks Neural (MANN)—Neural Turing Machine (NTM): (A) NTM architecture; (B) NTM general block diagram; (C) NTM time sequence RNN; and (D) NTM equations.

the “NN,” the “*Softmax layer*” takes as input a number of scores of values, and squashes them into values in the “range [0,1]” whose sum is “1.” It is used for “*multiclass classification*,” such as “*object recognition*”

- “*Location-based addressing*” is made in three steps: “*interpolation*,” “*convolution shift*,” and “*sharpening*” [39], where:
 - “*Interpolation*” is used to decide whether the weights from the previous time step “ $w_i(t-1)$ ” should be used or the actual weight obtained through “*content-based addressing*” in memory.
 - “*Convolution shift*” is used to move forward to the next weight position in memory but keeps track of where the last position is.
 - “*Sharpening*” is a value that indicates if sharpening is necessary or not; if it is needed because of the shift, the weight of a single location will be dispersed into other locations.

The “*Memory Augmented Neural Networks Neural (MANN): Neural Turing machine (NTM)*” typical sequence in time as a “*Recurrent Neural Network (RNN)*” is shown in Fig. 6.16C. Experiments demonstrate that besides “*NTM’s*” capability of learning simple algorithms, it is capable of generalizing beyond the training regime [40].

The “*Neural Turing Machine (NTM)*” is based on “*controllers*” and many different classes of “*Feed Forward Neural Networks types*” and “*Recurrent Neural Networks Types*” that can be used for this purpose. For example:

- “*Basic FFN types*” include two input nodes and one output node, as explained in Section 5.2.1 “*Perceptron (P) or Single-layer perceptron network*,” as shown in Fig. 5.2A.
- “*Multilevel feed forward network types*” approximate measurable function at any desired degree of accuracy and precision, as explained in Section 5.2.2 “*Multilayer Perceptron’s (MLP) also called Feed forward Neural Network (FFNN)*,” as shown in Fig. 5.4A.
- “*NARX neural network*” is a network that models the “*Turing machine*” and is able to simulate them by applying the “*sigmoid activation function*,” as explained in Section 5.3 “*Shallow neural network: Nonlinear autoregressive with External (Exogenous) Input (NARX)*” with a step by step example in Research 5.3.
- “*Recurrent Neural Networks*,” as explained in 6.2.1 “*Recurrent Neural Network (RNN) vanilla*,” takes the previous “*output or hidden*” states as “*inputs*,” as indicated in the “*RNN general network*” in Fig. 6.1B.
- “*Long/short-term memory (LSTM)*” is a special “*RNN*” capable of learning long-term dependencies, simulating in its feedback connections a “*general purpose computer*,” as explained in Section 6.2.2 and shown in Fig. 6.2.
- And many others.

Some examples of a “*Neural Turing Machine (NTM)*” in development are:

- “*Neural Turing Machines*” by Graves et al. [41]. They extend the capabilities of “*neural networks*” by coupling them to external memory resources, which they can interact with by attentional processes. The combined system is analogous to a “*Turing Machine or Von Neumann architecture*” but is “*differentiable end-to-end*,” allowing it to be efficiently trained with “*gradient descent*.” Preliminary results demonstrate that “*Neural Turing Machines*” can infer simple algorithms such as copying, sorting, and associative recall from input and output examples.
- “*A review on Neural Turing Machine*” by Faradonbeh and Safi-Esfahani [42]. One of the major objectives of “*Artificial Intelligence*” is to design learning algorithms like the human brain that are executed on general purpose computational machines. The “*Neural Turing Machine (NTM)*” is a step toward realizing such a computational machine. The attempt is made here to run a systematic review of “*Neural Turing Machines*.” First, the mind-map and taxonomy of machine learning, neural networks, and Turing machine are introduced. Next, “*NTM*” is inspected in terms of concepts, structure, variety of versions, implemented tasks, comparisons, etc. Finally, the paper discusses the issues and ends with several future works.
- “*Evolving Neural Turing Machines for Reward-based Learning*” by Greve et al. [43]. This is based on an unsolved problem in “*neuroevolution (NE)*” of evolving “*artificial neural networks (ANN)*” that can store and use information to change their behavior online. While “*plastic neural networks*” have shown promise in this context, they have difficulty retaining information over longer periods of time and integrating new information without losing previously acquired skills. They proposed an “*Evolvable Neural Turing Machine (ENTM)*” that can solve a simple copy task and, for the first time, the continuous version of the double T-Maze, a complex reinforcement-like learning problem. In the T-Maze learning task the agent uses the memory bank to display adaptive behavior that normally requires a “*plastic ANN*,” thereby suggesting a complementary and effective mechanism for adaptive behavior.

“*Neural Turing Machines*” show promising properties in terms of generalization compared to “*LSTMs*.” The “*NTMs*” open new possibilities to go a step further and understand how concepts are created. After all, algorithms are concepts based on instructions for computers. This is a new take on
(Continued)

(Continued)

“Artificial Intelligence: trying to teach machines things they can do, the same way we would learn them.” The way information is stored and modified locally, in an external memory, makes it an interesting solution to handle problems that require reasoning. That can be applied to develop many solutions for many problems in “Biomedical Engineering.”

6.3.2 Differentiable Neural Computers

“Differentiable Neural Computers (DNC)” use a form of “memory augmented neural network” and at the same time an “attention mechanism” that is a system of vectors and operations mediating between the controller and memory. The concept was originally defined as “Hybrid computing using a neural network with dynamic external memory” [44], that can learn using its memory to answer complex questions related to structured data, maps, trees, etc. The controller takes the role of a “Central Processing Unit (CPU)” through a “RNN,” where the “DNC can read/write from multiple locations in memory.” Memory can be “searched based on the content of each location,” or the “associative temporal links” can be followed forward and backward to recall information written in sequence or in reverse. The readout information can be used to produce answers to questions or actions to take in an environment [45]. The idea is to take the classical Von Neumann computer architecture and replace the CPU with an “RNN,” which learns when and what to read from the

RAM [46], as shown in Fig. 6.17A, and the “DNC” general block diagram shown in Fig. 6.17B.

Like in the “Neural Turing Machine (NTM),” the “memory” is a “ $N \times W$ matrix,” where each row is called a “location.” The matrix will have “ N locations, with W -dimensional vectors.” And the “attention mechanism” defines some distributions called “weightings” over the “ N locations.” The units in which the “weightings” are produced and used are known as “heads,” and there are two types of them: “read weighting” for the read process and “write weighting” for the writing process.

The big difference in “DNC” is the “differentiable attention mechanism” as shown in Fig. 6.17A, and its general block diagram in Fig. 6.17B, that is applied in three different ways to produce the “weightings” through a “interface vector,” emitted by the “controller as a set of values encoding its needs and request to be made for the memory,” these request process are: “content lookup,” “temporary memory linkage,” and “dynamic memory allocation” as shown in Fig. 6.17C, where:

- “Content lookup” is a comparative process in each location, such as a “similarity measurement,” applying the “cosine similarity concept” as shown in Fig. 6.16D eq. 3).
- “Temporary memory linkage” are consecutive written locations in memory, keeping track of where the process begins writing and where the process finishes writing in the memory matrix.
- “Dynamic memory allocation”: each location has a usage level from “0” to “1,” which indicates if it is an unused location that is sent to the “write head” to know

Memory Augmented Neural Networks Neural (MANN): Differentiable Neural Computers (DNC)

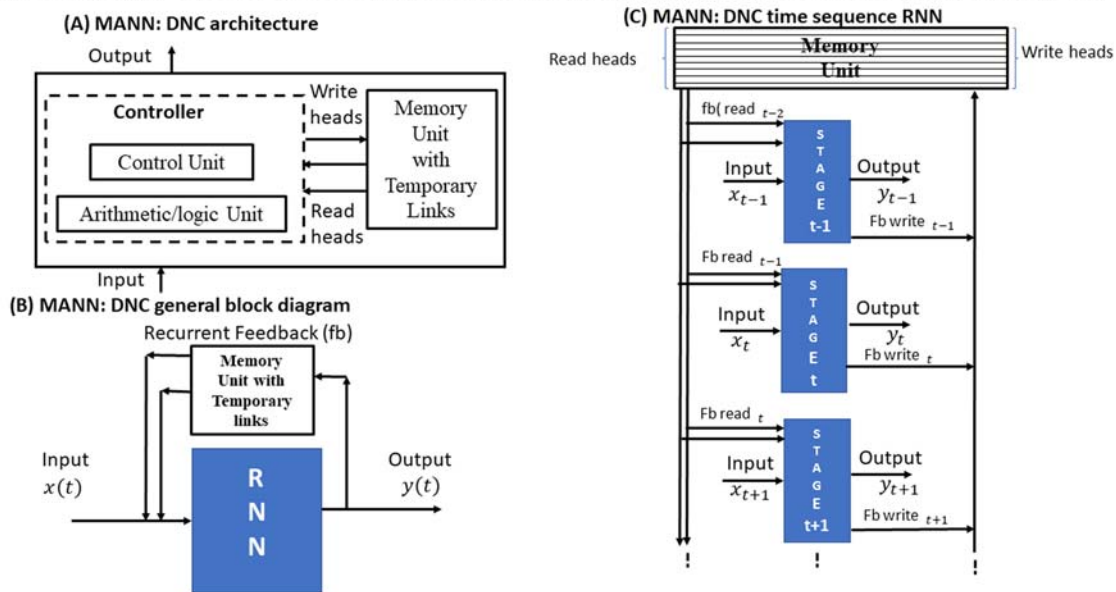


FIGURE 6.17 Differentiable Neural Computers (DNC): (A) DNC architecture, (B) DNC general block diagram, and (C) DNC time sequence RNN.

where to store new information without erasing old information. Besides, the “dynamic memory allocation” refers to the ability of the controller to relocate the memory that is no longer required, erasing its content.

Some “Differentiable Neural Computers (DNC)” examples that could be used in “Biomedical Engineering” are:

- “Employing Differentiable Neural Computers for Image Captioning and Neural Machine Translation” by Sharma et al. [47]. In this paper the problem of language translation by leveraging the capabilities of the recently developed “DNC architectures” is applied. They modified the “DNC typical architecture” by including “dual neural controllers” instead of one, and an “external memory module.” Inside the controller is a “neural network with memory-augmentation” which differs from the original “differentiable neural computer,” they implemented a dual controller’s system in which one controller is for encoding the query sequence whereas another controller is for decoding the translated sequences.
- “EDNC: Evolving Differentiable Neural Computers” by Rasekh and Safi-Esfahani [48]. They used the approach that includes methods for choosing the controller’s structure and weight optimization, like “NeuroEvolution” that automatically, and without prior knowledge, obtains an appropriate network structure. This research presents “Evolutionary Differentiable Neural Computer (EDNC),” which uses a novel “NeuroEvolution algorithm” that is introduced in two types of nested object-oriented encoding called “Adaptive Layer NeuroEvolution (ALNE)” and a “Matrix-based one called “M_ALNE.” These “NeuroEvolution algorithms” use the following subalgorithms: “Self-Adaptivity in the Initial Population (SAIP),” “Self-Adaptivity in Evolutionary Structures (SAES),” “Layer Recombination (LR),” “Layer Mutation (LM),” and “Self-Adaptivity in Mutation (SAM).” The evolution process starts with “SAIP,” and then it takes short and long steps to explore a variety of neural structures by a “SAES algorithm” that uses both “LR and LM algorithms” to provide a variety of structures. Finally, the “SAM algorithm” guides the selected structures to a target structure.

6.3.2.1 Research 6.6 simulation of a Turing Machine (TM) using a recursive function

6.3.2.1.1 Case for research

“Obtain a simulation of a Turing Machine (TM) using a recursive function to analyze how the respiratory transmission of a COVID-19 infected person can spread a virus through droplets/mini-droplets emissions.”

6.3.2.1.2 General objective

Apply “MATLAB” to define, build, and make “a simulation of a Turing Machine (TM) model using a recursive function to analyze how the respiratory transmission of

an COVID-19 infected person can spread a virus through droplets/mini-droplets emissions, specifying different variables, such as number of droplets in each respiration cycle, rate of respiration, and wind speed at exhalation, to obtain a result during the specified time, with the main objective of applying appropriate infection control and safe distance measures to slow its spread.”

6.3.2.1.3 Specific objectives

- Define initial parameters for the simulation “Turing Machine as Recursive Function (TMRF).”
- Define external parameters for the simulation of “Neural Turing Machine (NTM).”
- Run simulation of “TMRF.”
- Apply recursion of a function to simulate the “Turing Machine” as a “Neural Turing Machine (NTM).”
- Count the number of air droplets in different specific areas.

6.3.2.1.4 Background for “Modes of transmission of SARS-CoV-2 (COVID-19)”

The two most common causative agents of infectious disease are the “virus” and “bacterium.” “Viruses are not living organisms, bacteria are.” Viruses only grow and reproduce inside of the host cells they infect. When found outside of these living cells, viruses are dormant. Their “life” therefore requires the hijacking of the biochemical activities of a living cell. Bacteria, on the other hand, are living organisms that consist of single cells that can generate energy, make their own food, move, and reproduce (typically by binary fission). “Bacteria” are giants when compared to viruses. The smallest bacteria are about 0.4 μm (one millionth of a meter) in diameter while viruses’ range in size from 0.02 to 0.25 μm . This makes most viruses submicroscopic, unable to be seen in an ordinary light microscope; they are typically studied with an electron microscope. Current evidence suggests that “SARS-CoV-2,” the virus that causes “COVID-19,” is predominantly spread from person-to-person. Understanding how, when, and in what types of settings “SARS-CoV-2” spreads is critical to develop effective public health and infection prevention and the control measures to break chains of transmission [49]. The most common modes of “SARS-CoV-2” virus transmission are “contact,” “droplet,” “airborne,” “fomite,” “fecal-oral,” “bloodborne,” “mother-to-child,” and “animal-to-human transmission,” where:

- “Contact”: “SARS-CoV-2” is transmitted very efficiently via “direct contact” of an infected human with other humans during their daily routines.
- “Droplet”: “SARS-CoV-2” is transmitted also very efficiently “via respiratory droplets and/aerosols” from 1 to 7 days after exposure. The term “droplet”

refers mostly to water with various inclusions, depending on how it is generated. Humans naturally produce droplets by “breathing,” “talking,” “sneezing,” “coughing,” etc. “Droplets” are of various sizes and include various cells types, such as epithelial cells and cells of the immune system, physiological electrolytes contained in mucous and saliva, such as “sodium ion (Na^+)”, “potassium ion (K^+)”, “chlorine gas (Cl^-)” but also various potentially infectious agents, such as “bacteria,” “fungi,” and unfortunately “viruses,” such as the “SARS-CoV-2.” Currently, the term “droplet is often taken to refer to droplets $>5 \mu m$ in diameter that fall rapidly to the ground under gravity over a limited distance usually $\leq 1 m$.” In contrast, the term “droplet nuclei or mini-droplet refers to droplets $\leq 5 \mu m$ in diameter that can remain suspended in air for significant periods of time, allowing them to be transmitted over distances $>1 m$ ” [50].

- “Airborne”: “SARS-CoV-2 airborne transmission” refers to the passage of a microorganism from a source to a person through aerosols resulting in infection.
- “Fomite”: “SARS-CoV-2 fomite transmission” refers to the transmission of infectious disease by objects.
- “Fecal-oral”: “SARS-CoV-2 fecal-oral transmission” refers to the different pathways of the infectious diseases from fecal to oral, such as fecal–hands–food–mouth, fecal–air–objects–mouth, fecal–hands–water–food–mouth, etc.
- “Bloodborne”: “SARS-CoV-2 bloodborne transmission” refers to the transmission of disease when an infected people person’s body fluids somehow get inside another person’s body
- “Mother-to-child”: “SARS-CoV-2 mother-to-child transmission” occurs through different paths, such as “intrauterine transmission” of viral transmission from mother to fetus, “intrapartum transmission” during or directly after delivery, and “transient viremia” as a superficial exposure [51].
- “Animal-to-human transmission or vice versa”: “SARS-CoV-2” originally jumped from animals to humans. There is now abundant evidence that humans can transmit “SARS-CoV-2” to domestic pets and other animals. The more people infected the greater the risk of creating a permanent animal reservoir of infection; one that may pose a threat for generations.

The main objective of this research is to simulate how the respiratory transmission of an “SARS-CoV-2” infected person can spread a virus through “droplets/mini-droplets emissions,” specifying different variables, such as number

of droplets in each respiration cycle, rate of respiration, and wind speed at exhalation, to obtain a result during a specific time, with the main objective of applying appropriate infection control measures and to slow its spread [52].

6.3.2.1.5 Dataset

There is not a specific dataset for this research, it is a simulation of “Droplet/mini-droplet” generated by the human breath based on different variables, such as “number of droplets in each respiration cycle,” “rate of respiration,” “exhalation wind speed,” “time for analysis,” and others:

- “Number of droplets/mini-droplets in each respiration cycle”: this depends on different factors, such as anatomy and physiology of respiratory system, body humidity, total droplet weight, grade of infection, and many more. Any value of droplets/mini-droplets can be specified from 30 to an any specific reasonable max value can be analyzed and represented on this algorithm. This value represent the number of “droplet nuclei or mini-droplets, referring to droplets $\leq 5 \mu m$ in diameter, that can remain suspended in air for significant periods of time, allowing them to be transmitted over distances.” “Respiration rate” is a numeric value based on “normal respiration rates for an adult person at rest range from 12 to 16 breaths per minute” [53].
- “Exhalation wind coefficient” is based on the flow of the breath out of a human being and the elasticity of their lungs based on the internal surface of the lungs; “on average a nonemphysemic person normally can hold about 5 L of air volume” [54]; an estimation from 10 to higher coefficient values can be used in the simulation.
- “Time for analysis” any value in minutes can be used for the simulation, the default value of 10 minutes was used to allow enough time for the mini-droplets to distribute beyond the original limits of $400 \times 400 \times 400$ cm, that is, a social distance of 2 m.

The main objective is to obtain an analysis over a specific time in order to “apply appropriate infection control safe distance measures to slow its spread,” as shown in Fig. 6.18.

6.3.2.1.6 Procedure

The MATLAB simulation is summarized in Table of slides 6.6, and each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

Simulation “Neural Turing Machine (NTM)”

The main objective of this research is to “Analyze how the respiratory transmission of a virus infected person can sparse a virus through his droplets/mini-droplets emissions” to apply appropriate infection control measures and to slow down its spread

Simulation: “Droplet/mini droplet” generated by the human breath based on different variables as:

- Number of droplets in each respiration cycle
- Rate of respiration
- Exhalation wind speed
- Time for analysis
- Room size 200 x 200 x 200 cm

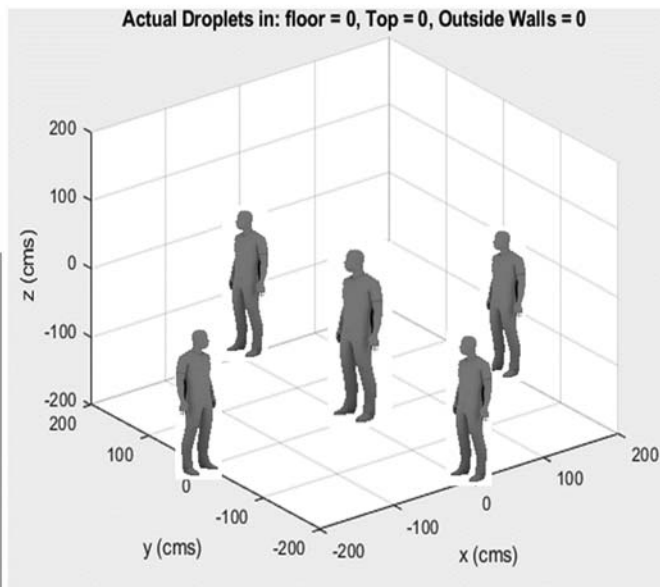


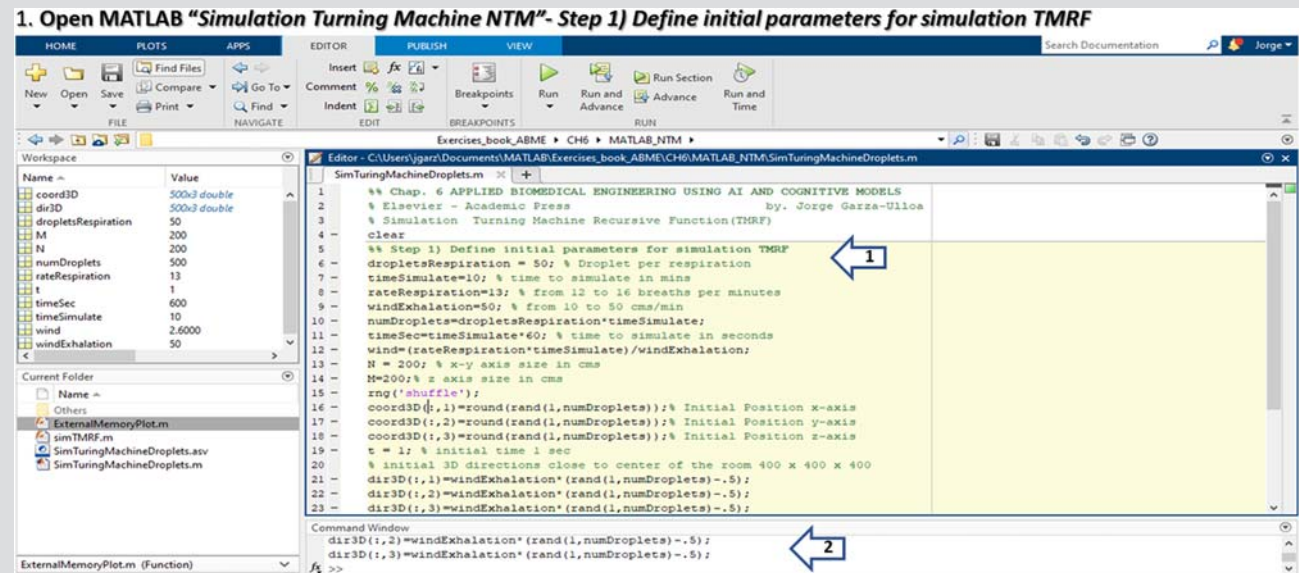
FIGURE 6.18 Simulation of NTN to “analyze how the respiratory transmission of a SARS-CoV-2 virus-infected person can spread a virus through droplets/mini-droplets emissions.”

Table of slides 6.6 Steps for MATLAB to obtain a simulation of a “Turing Machine (TM) using a recursive function (RF) simulating a RNN to analyze how a respiratory virus-infected person can infect the space through droplets/mini-droplets”.

Slide 1 Description Screen figure

1 Open MATLAB “Simulation Turing Machine NTM”—Step 1) Define initial parameters for simulation Turing Machine Recursive Function (TMRF)

Go to the directory “... \Exercises_book_ABME\CH6\MATLAB_NTM.” Open the script “SimTuringMachineDroplets.m.” Copy and paste in the command prompt: “Step 1) Define initial parameters for simulation TMRF”. The variables for the calculation for “Droplets/mini-droplets” are set to be analyzed during 10 min in a space of “400 × 400 × 400 (cm),” where the virus-infected person is at the center



Go to the directory “... \Exercises_book_ABME\CH6\MATLAB_NTM”. Open the script “SimTuringMachineDroplets.m”. Copy and paste in the command prompt: “Step 1) Define initial parameters for simulation TMRF”. The variables for the calculation for “Droplets /mini-droplets” are set to be analyzed during 10 minutes in a space of “400x 400 x 400 (cm)”, where the virus infected person is at the center .

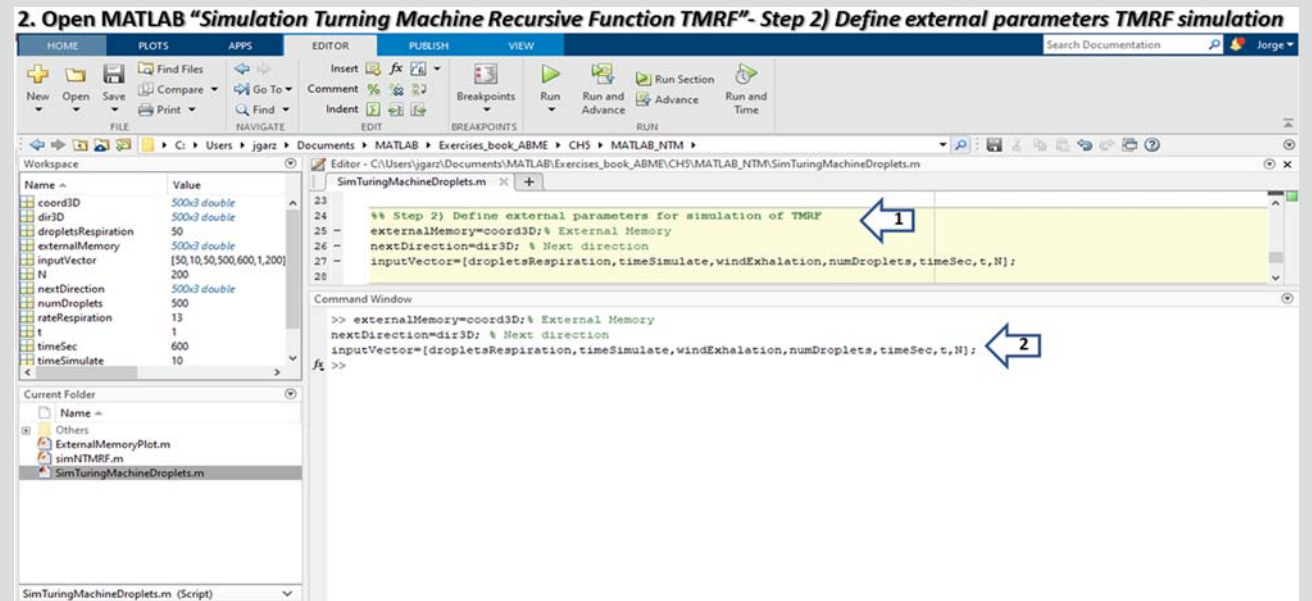
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 2 Description Open MATLAB “Simulation Turing Machine Recursive Function TMRF”—Step 2) Define external parameters TMRF simulation Copy and paste in the command prompt: “Step 2) Define external parameters for simulation of NTM.” All the information needed is summarized in 3 variables: “inputVector” as a vector with 7 elements, “nextDirection” as incremental droplet direction in matrix of “numDroplets × 3,” and “externalMemory” as 3D coordinates with matrix “numDroplets × 3”

Screen figure



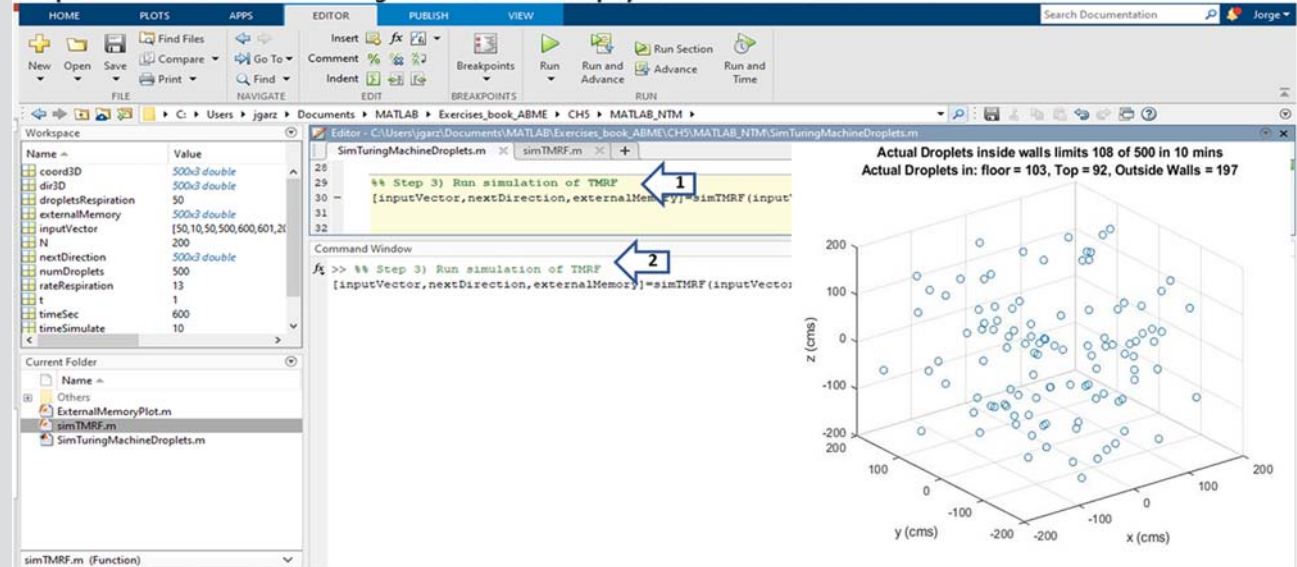
Copy and paste in the command prompt: “Step 2) Define external parameters for simulation of NTM”. All the information needed is summarized in 3 variables: “inputVector” as a vector with 7 elements, “nextDirection” as incremental droplet direction in matrix of “numDroplets x 3”, and “externalMemory” as 3D coordinates with matrix “numDroplets x 3”.

3

Open MATLAB "Simulation Turing Machine TMRF" — Step 3) Run NTMRF simulation

Copy and paste in the command prompt: "Step 3) Run simulation of TMRF." The simulation will show step by step how the droplets are expanded in the 3 axes, showing statistics for every second of how many droplets are: "inside walls (axis limits)," "fall to the floor," "exit in the top," and "number of droplets going out of the walls in the time specified." It called two MATLAB user functions as shown in the next two slides

3. Open MATLAB "Simulation Turing Machine TMRF"- Step 3) Run NTMRF simulation



Copy and paste in the command prompt: "Step 3) Run simulation of TMRF ". The simulation will show step by step how the droplets are expanded in the 3 axis, showing statistics por every second of how many droplets are: "inside walls (axis limits)", "fall to the floor", "exit in the top" and "number of droplets going out of the walls in the time specified". It called two MATLAB user functions as shown in the next two slides.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

4 MATLAB “Simulation Turing Machine NTMRF”—User function “simTMRF()”
The simulation of the “Turing Machine” is executed by this “recursive function” that update of the parameters needed. The most important is that automatically keep track of “Location-based addressing” and the “External Memory update,” these steps to obtain the parameters updated are necessary to build a “Neural Turing Machine” as shown in their NTM equations at Fig. 6.16

4. MATLAB “Simulation Turing Machine NTMRF”— user function simTMRF()

```
1 %% Chap. 5 APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
2 %% Elsevier - Academic Press by. Jorge Garza-Ulloa
3 %% Simulation Recursive Machine based on recursive function
4 function [inputVector,nextDirection,externalMemory]=simTMRF(inputVector,nextDirection,externalMemory)
5 % set new droplets positions
6 externalMemory(:,1)=externalMemory(:,1)+(round(2*rand()-1)*nextDirection(:,1));
7 externalMemory(:,2)=externalMemory(:,2)+(round(2*rand()-1)*nextDirection(:,2));
8 externalMemory(:,3)=externalMemory(:,3)+(round(2*rand()-1)*nextDirection(:,3));
9 % 3D plot
10 ExternalMemoryPlot(inputVector,externalMemory);
11 inputVector(6)=inputVector(6)+1; % next time
12 for number = 1:inputVector(4) % inputVector(4)=numDroplets
13     if inputVector(6) <= inputVector(5)
14         nextDirection(:,1)=inputVector(3) * (rand(1,inputVector(4))-0.5);
15         nextDirection(:,2)=inputVector(3) * (rand(1,inputVector(4))-0.5);
16         nextDirection(:,3)=inputVector(3) * (rand(1,inputVector(4))-0.5);
17         % Call recursive function NTMRF
18         [inputVector,nextDirection,externalMemory]=simTMRF(inputVector,nextDirection,externalMemory);
19     else
20         return % Exit recursive function NTMRF
21     end
22 end
```

Command Window
>> SimTuringMachineDroplets

The simulation of the “Turing Machine” is executed by this “recursive function” that update of the parameters needed. The most important is that automatically keep track of “Location-based addressing” and the “External Memory update,” these steps to obtain the parameters updated are necessary to build a “Neural Turing Machine” as shown in their NTM equations at figure 6.16.
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

5

“Simulation Turing Machine NTMRF”—user function “ExternalMemoryPlot()”

This function is called from the function “simTMRF,” to calculate the droplets inside and outside of the walls (axis limits)

5. “Simulation Turing Machine NTMRF”—user function ExternalMemoryPlot()

```
1 %% Chap. 5 APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS
2 % Elsevier - Academic Press
3 % 3D Plot for Simulation Recursive Machine based on recursive function
4 function ExternalMemoryPlot(inputVector,externalMemory)
5 - countFloor=sum(externalMemory(:,3)-inputVector(:,7)); % ← 1
6 - countTop=sum(externalMemory(:,3)>inputVector(:,7)); %
7 - countxyz=sum(externalMemory(:,1)<inputVector(:,7) & (externalMemory(:,1)>inputVector(:,7) & (externalMemory(:,2)<inputVector(:,7) & (externalMemory(:,2)>inputVector(:,7)));
8 - countWalls=0;
9 - if (countFloor>0 && countTop>0)
10 - countWalls=inputVector(4)-countxyz-countFloor-countTop;
11 - end
12 - stem3(externalMemory(:,1),externalMemory(:,2),externalMemory(:,3),'linestyle','none');
13 - title([
14 - ['Actual Droplets inside walls limits ' num2str(countxyz) ' of ' num2str(inputVector(4)) ' in ' num2str(inputVector(6)/6) '
15 - ['Actual Droplets in: floor = ' num2str(countFloor) ', Top = ' num2str(countTop) ', Outside Walls = ' num2str(countWalls)
16 - ]);
17 - xlim([-inputVector(7),inputVector(7)]); ← 2
18 - ylim([-inputVector(7),inputVector(7)]);
19 - zlim([-inputVector(7),inputVector(7)]);
20 - xlabel(" x (cms)");ylabel(" y (cms)");zlabel(" z (cms)");
21 - pause(.01);
```

This function is called from the function “simTMRF”, to calculate the droplets inside and outside of the walls (axis limits).

Conclusions

- “The 2-m social distancing rule is based on an outdated model from other viruses created some time ago 2006/2007 [55], so it needs analysis of the complex transmission of a continuous transmission of droplets of different sizes and exhaled air shapes in terms of the range that they can reach.”
- “Smaller airborne droplets with SARS-CoV-2 may spread up to 8 m concentrated in exhaled air from infected individuals, even without background ventilation or airflow.”
- “The risk of SARS-CoV-2 transmission falls as physical distance between people increases, so relaxing the distancing rules, particularly for indoor settings, might therefore risk an increase in infection rates. In some settings, even 2 m may be too close.”
- “Safe transmission mitigation measures depend on multiple factors” such as “viral load,” “duration of exposure,” “number of individuals,” “indoor versus outdoor settings,” “level of ventilation, and whether face coverings are worn.”, etc.

Recommendation

- The “2-m social distance for SAR-CoV-2 is not enough,” and sometimes it is even too close when air circulation exists. This simulation shows that the 100% recommended safe distance is 4–5 m.
- The use of “masks for all the people is a must” to diminish the spread of the virus.
- The “time exposure” and the “grade of infection in people around us” are two factors that must be considered to stop the spread of the virus.
- The “fomite” accumulation on the floor and objects around an infected people is a big and important issue but it is determined on how long the virus can survive.
- “Social distancing for SARS-CoV-2” should be updated and used alongside other strategies to reduce transmission, such as air hygiene, involving in part maximizing and adapting ventilation to specific indoor spaces, air conditioning with UV light to fight COVID-19 spread [56], effective hand washing, regular surface cleaning, face coverings where appropriate, prompt isolation of affected individuals and others.

6.4 Modular Neural Networks types

Modular Neural Networks implies the use of a collection of different networks working independently or dependently and contributing toward the output. Each neural network has a set of inputs, which are unique compared to other networks, constructing and performing subtasks. If they are independent, these networks do not interact or signal with each other in accomplishing the tasks and can break down a large computational process into smaller components decreasing the complexity. This breakdown will help in decreasing the number of connections and

negates the interaction of these networks with each other, which in turn will increase the computation speed. If the networks are dependent, one network complements the work of the other. Some examples are shown in Fig. 1.11, that is, *Deep Belief Network (DBN)* and others.

6.4.1 Deep Belief Network

“*Deep Belief Network (DBN)*” is a generative graphical representation because it produces all possible values that can be generated based on “*probability and statistics.*” “*DBN*” is composed of multiple layers with values, where there is a special relation between its layers but not with their unbiased values to be stored in leaf nodes. “*DBN*” is composed of unsupervised networks like the “*Restricted Boltzmann Machines (RBMs)*” network for the pretrain phase, where the invisible layer of each subnetwork is the visible layer of the next. The hidden or invisible layers are not connected to each other and are conditionally independent and then there is a *Feed Forward Network* for the fine-tune phase, in which each “*RBM*” layer communicates with the previous and subsequent layers, but the nodes of any single layer do not communicate with each other laterally. “*DBN*” can be used as a classifier, or if ending with a “*Softmax* layer*” as unsupervised learning. “*DBN*” are used to recognize, cluster, generate images, video sequences, and motion-capture data.

Note*: *Softmax function takes an input as a vector of K real numbers and normalizes it into a probability distribution consisting of K probabilities. The input vectors that could contain negative or greater than 1 values are converted to components in the interval (0,1), and they will add up to 1, representing probability values.*

The first step in “*DBN*” is the training of a layer of properties that can obtain the input signals from the pixels directly. Then, the second step is to treat the values of this layer as pixels to learn features of the previously obtained features in its second hidden layer, as shown in Fig. 6.19A, and every time that another layer of properties or features is added to the “*belief network*” there is an improvement in the lower bound on the log probability of the training dataset, as indicated in Fig. 6.19B as “*DBN Step 2*” and Fig. 6.19C as “*DBN Step 3.*” The “*greedy learning algorithm**” applies the training of each “*RBM*” at a time, and until all the “*RBMs*” have been trained.

Note: The “*greedy learning algorithm**” is any algorithm that follows the problem-solving heuristic of making the locally optimal choice at each stage [57].

Some examples of “*Deep Belief Network (DBN)*” applied to “*Biomedical Engineering*” are:

- “*Composite Deep Belief Network approach for enhanced Antepartum fetal electrocardiogram signal*” by Jagannath et al. [58]. “*Antepartum Fetal*”

surveillance is the most vital area of investigation during the pregnancy period. The “*fetal electrocardiogram (fECG)*” signal can be detected during pregnancy from the “*antepartum stage.*” Generally, “*fECG*” signal analysis is not carried out for “*Fetal surveillance.*” Rather, the traditional methodologies like “*phonocardiogram,*” etc. are utilized. The reason is the unavailability of an effective methodology for providing a good quality fECG signal. The proposal of a hybrid tactic called “*Bayesian Deep Belief Network (BDBN) for fECG*” signal enhancement is presented in this article. The proposed BDBN technique involves “*Bayes’ filtering methodology in amalgamation with Deep Belief Network.*” The Bayes’ filtering was employed to eliminate undesired signal components. A deep learning (DL) technique was utilized with Deep Belief Network (DBN) to extract high-quality fECG signal. The methodology resulted in a good quality fECG signal which is indeed valuable for timely physician analysis.

- “*Deep belief network and linear perceptron based cognitive computing for collaborative robots*” by Lv and Qiao [59]. This paper has the objective to analyze the performance of the control system of collaborative robots based on “*cognitive computing technology*” using a combination of “*cognitive computing*” and “*deep belief network*” algorithms with collaborative

robots to construct a cognitive computing system model based on deep belief networks, which is applied to the control system of collaborative robots. Further, the simulation is used to compare and analyze the algorithm performance of a “*deep belief network (DBN),*” “*multilayer perceptron (MLP),*” and the “*cognitive computing*” system model of a “*deep belief network and linear perceptron (DBNLP),*” reaching the conclusion that the application of the “*DBNLP*” algorithm model to collaborative robots can significantly improve their accuracy and safety, providing an experimental basis for the performance improvement of later collaborative robots.

- “*Modeling task-based fMRI data via deep belief network with neural architecture search*” by Qiang et al. [60]. In this paper they proposed an unsupervised “*neural architecture search (NAS) framework*” on a “*deep belief network (DBN)*” that models volumetric “*Functional magnetic resonance imaging (fMRI)*” data, named “*NAS-DBN.*” The “*NAS-DBN framework*” is based on “*Particle Swarm Optimization (PSO),*” where the swarms of neural architectures can evolve and converge to a feasible optimal solution. The experiments showed that the proposed “*NAS-DBN*” framework can quickly find a robust architecture of “*DBN,*” yielding a hierarchy organization of functional brain networks (FBNs) and temporal responses.

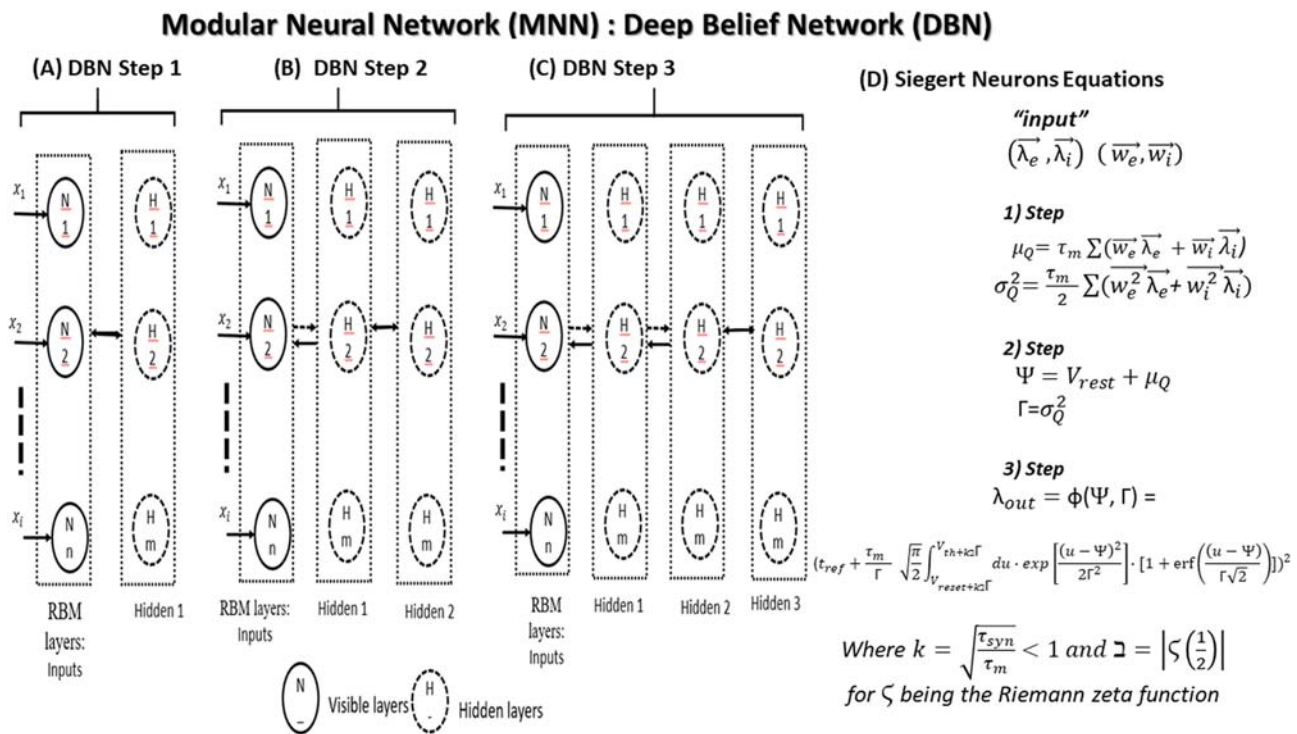


FIGURE 6.19 Deep Belief Network (DBN) based on RBM layers: (A) DBN Step 1, (B) DBN Step 2, (C) DBN Step 3, and (D) Siegert neurons equations.

In the example shown in the Research 6.7 a “DBN” [61] is a “*Spiking deep belief network (SDBN)*” that is constructed by stacking “RBM”s and interpreting the hidden layer of the lower “RBM” as the visible layer, known as the “*visual input layer*,” with 784 units from images with “ 28×28 pixel” resolution, to the next hidden layer known as the “*visual abstraction layer*” that has the visualization of the weight learned as a subset. Because the “RBM” is a “*Recurrent Neural Network (RNN)*” in the recursive process the “*Visual Input Layer*” can be seen in two ways: “*Visual Input Layer Bottom-Up*” and “*Visual Input Layer Top-Down*.” And finally, an “*Associative layer*” of 10 units for classification is built to identify the image in the categories that must agree with the specific label in the training. The Research 6.7 is an example of a “*Neural simulation*” based on “*spiking generative modes*” applying the “*leaky integrate-and-fire neurons (LIF)*” known as “*Siebert neuron**” that can be used in Poisson input trains by analyzing the subthreshold activity of the neurons.

Note: “*Siebert Neurons*” have outputs that approximate the mean firing rate of leaky integrate-and-fire neurons with the same parameters using the sequence of equations shown in Fig. 6.19D.

6.4.1.1 Research 6.7 Create a Deep Belief Network model to analyze and differentiate normal and pneumonia chest X-rays

6.4.1.1.1 Case for research

Obtain a “*DBN model*” from MATLAB based on node-neurons from “*Spiking Neural Networks (SNN)*” applying the “*Siebert Neurons mathematical model*” to “*differentiate normal and pneumonia chest X-rays*.”

6.4.1.1.2 General objective

Apply “*MATLAB*” to define, build, and make “*a simulation of a Deep belief Network model*” using “*Recurrent Neural Network (RNN)*” as the “*Restricted Boltzmann Machine (RBM)*” using the “*greedy algorithm*” and applying the “*Siebert Neurons mathematical model*” for node-neurons

firing as a “*Spiking Neural Networks (SNN)*,” with the objective of obtaining a “*classification of chest X-ray images to differentiate normal and pneumonia*.”

6.4.1.1.3 Specific objectives

- “*Load all chest X-ray images*” from the two categories in an image store MATLAB variable and split them as 70% for training, and 30% for validation (testing).
- “*Prepare train and validation chest X-rays*,” and label them as a vector, organized as a matrix.
- “*Setup all net variables and train DBN-network*” to finish visualizing all images generated.
- “*Show the classification process of the DBN live*” to test “*Chest X-ray images*” and find the correct category in the classification.

6.4.1.1.4 Background for “COVID-19 lung imaging chest X-rays

Note: Please read Research 6.5, section 6.2.10.1 background for “*COVID-19 lung imaging chest X-rays*.”

6.4.1.1.5 Dataset

The dataset images are in a folder named “*Chest_X_Rays*” that has two subfolders for general categories, as shown in Fig. 6.20:

- Subfolder “1” for “*Normal Chest X-rays*” with 12 CXR images.
- Subfolder “2” for “*COVID-19 Pneumonia Chest X-rays*” with 9 CXR images.

Note: This images dataset is available in the companion directory of the book, in the following folder “*..\Exercises_book_ABME\CH6\MATLAB_DBN*.”

6.4.1.1.6 Procedure

The MATLAB simulation is summarized in *Table of slides 6.7*, and each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

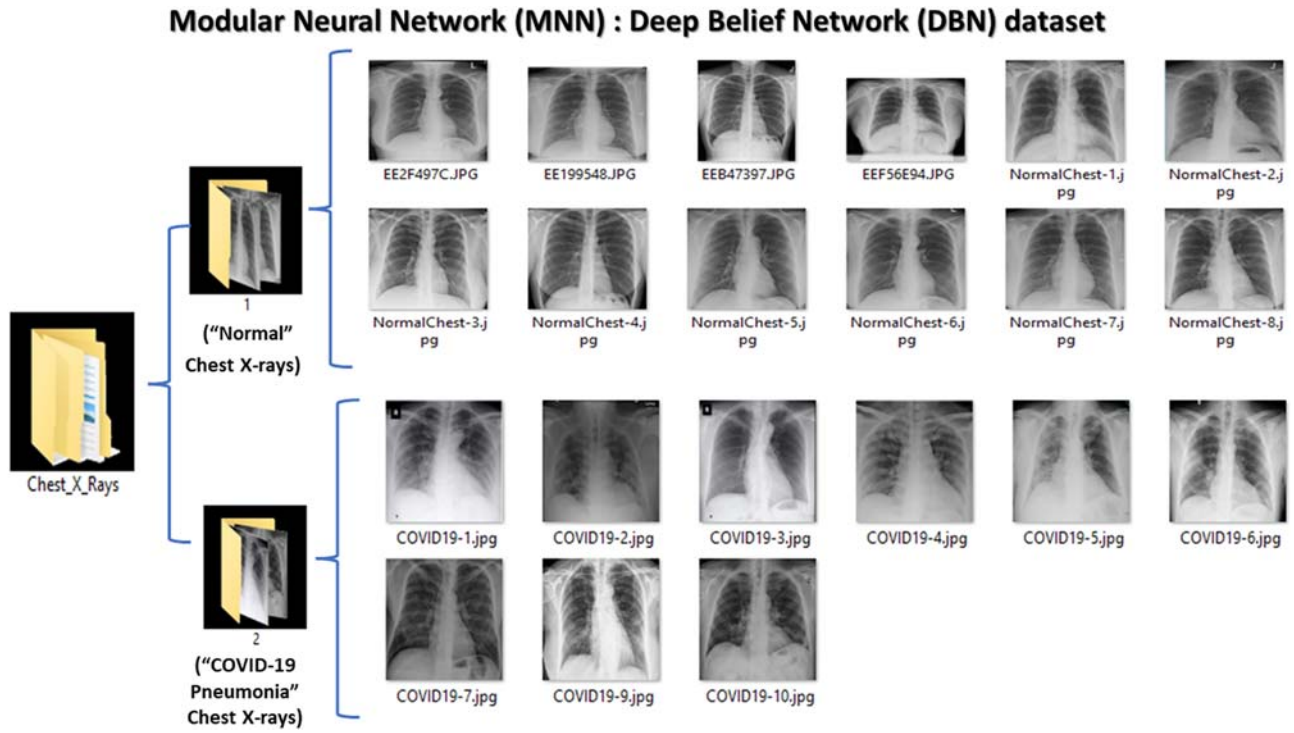
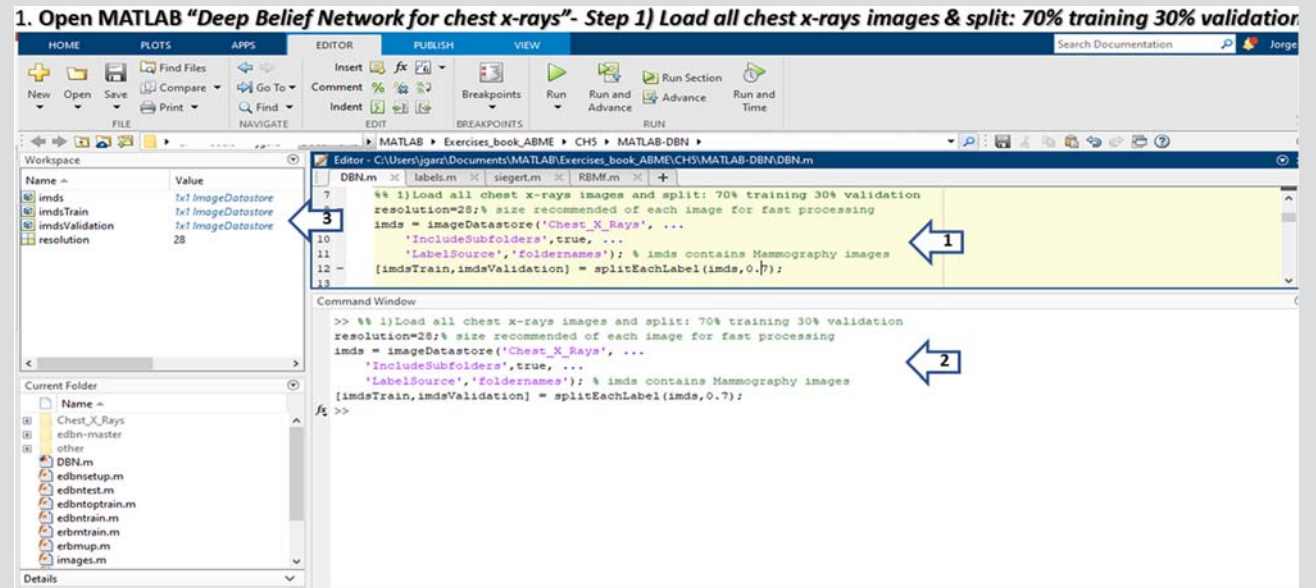


FIGURE 6.20 Deep Belief Network (DBN) dataset with two subfolders: "1 (Normal Chest X-rays)", and "2 (COVID-19 Pneumonia Chest X-rays)".

Table of slides 6.7 Steps for MATLAB to obtain an DBN model from MATLAB based on node-neurons from Spiking Neural Networks (SNN) applying the “Siegert Neurons mathematical model to differentiate normal and pneumonia chest X-rays.

Slide 1 Description Screen figure

1 Open MATLAB “Deep Belief Network for chest X-rays”—Step 1) Load all chest X-ray images and split: 70% training 30% validation
Go to the directory “... \Exercises_book_ABME\CH6 \MATLAB-DBN.” Open the script “DBN.m.” Copy and paste in the command prompt: “Step 1) Load all chest X-ray images and split: 70% training 30% validation.” Where the original images are read on an imagestore MATLAB variable and randomly divide in training and validation dataset



Go to the directory “... \Exercises_book_ABME\CH6\MATLAB-DBN”. Open the script “DBN.m”. Copy and paste in the command prompt: “Step 1) Load all chest x-rays images and split: 70% training 30% validation”. Where the original images are read on an imagestore MATLAB variable and randomly divide in training and validation data set.

2

MATLAB “Deep Belief Network for chest X-rays”—Step 2) Prepare Train and Validation images and labels as a vector

Copy and paste in the command prompt: “Step 2) Prepare Train and Validation images and labels as a vector.” For this example, the two MATLAB user functions as called: “labels() and images()” that read the label and images from the subdirectory classes and generate a vector place in matrices: “images_x_T” and “images_y_T” for Training and “images_x_V” and “images_y_V” for validation as seen in the inserted Figures

2. MATLAB “Deep Belief Network for chest x-rays”- Step 2) Prepare Train and Validation images and labels as a vector

The screenshot shows the MATLAB IDE interface. The workspace on the left lists variables: `images_x_T` (14x784 double), `images_x_V` (7x784 double), `imds` (1x1 ImageDatastore), `imdsTrain` (1x1 ImageDatastore), `imdsValidation` (1x1 ImageDatastore), `label_y_T` (14x10 double), `label_y_V` (7x10 double), `n_T` (14), `n_V` (7), and `resolution` (28). The Command Window shows the execution of the following code:

```
>> % 2) Prepare Train and Validation images and labels as a vector
[n_T,label_y_T]= labels(imdsTrain);
[images_x_T]= images(imdsTrain,n_T,resolution);
disp('Training images for Normal Chest X-Rays... ready');
[n_V,label_y_V]= labels(imdsValidation);
[images_x_V]= images(imdsValidation,n_V,resolution);
disp('Testing images for abnormal Chest X-Rays... ready');
```

The Command Window also displays two sets of chest X-ray images. The first set is labeled "Random Chest X-Rays Data set" and "Training set", showing 7 images. The second set is labeled "Random Chest X-Rays Data set" and "set", showing 14 images. Numbered arrows (1-4) point to the code, workspace, and image displays.

Copy and paste in the command prompt: “Step 2) Prepare Train and Validation images and labels as a vector”. For this example, the two MATLAB user functions as called : “labels() and images()” that read the label and images from the subdirectory classes and generate a vector place in matrices: “images_x_T” and “images_y_T” for Training and “images_x_V” and “images_y_V” for validation as shown in the inserted figures. From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

3 MATLAB "Deep Belief Network for chest X-rays"—Step 3) Setup and Train DBN-network and visualize images generation
Copy and paste in the command prompt: "Step 3) Setup and Train DBN-network and visualize images generation." The training is made in 150 epochs, because the weights are randomly generated in the first step please verify that it has scored ≥ 75 , if not please repeat step 3

3. MATLAB "Deep Belief Network for chest x-rays"- Step 3) Setup and Train DBN-network and visualize images generation

The screenshot shows the MATLAB IDE with the following components:

- Workspace:** Lists variables such as `edbn` (1x1 struct), `er` (0.1429), `images_x_T` (14x784 double), `images_x_V` (7x784 double), `imds` (1x1 ImageDatastore), `imdsTrain` (1x1 ImageDatastore), `imdsValidation` (1x1 ImageDatastore), `label_y_T` (14x10 double), `label_y_V` (7x10 double), `n_T` (14), `n_V` (7), and `opts` (1x1 struct).
- Command Window:** Shows the training progress:

```
Epoch 140: mean error: 0.22030.  
Epoch 141: mean error: 0.33120.  
Epoch 142: mean error: 0.28002.  
Epoch 143: mean error: 0.25697.  
Epoch 144: mean error: 0.21094.  
Epoch 145: mean error: 0.27223.  
Epoch 146: mean error: 0.20777.  
Epoch 147: mean error: 0.43674.  
Epoch 148: mean error: 0.20573.  
Epoch 149: mean error: 0.18102.  
Epoch 150: mean error: 0.26785.  
Scored: 85.71
```
- Figure Window:** Titled "Deep Belief Network Generated images Chest X-Rays", showing a 5x5 grid of 25 generated chest X-ray images.

Copy and paste in the command prompt: "Step 3) Setup and Train DBN-network and visualize images generation". The training is made in 150 epochs, because the weights are randomly generated in the first step please verify that it has scored ≥ 75 , if not please repeat step 3.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

4

MATLAB “Deep Belief Network for chest X-rays”—Step 4) DBN live to test 2 Chest X-ray images

Copy and paste in the command prompt: “Step 4) DBN live to test 2 Chest X-ray images.” Where the test images are analyzing showing the dynamic spikes and then its histogram is generated to see the class (category) obtain in the classification. *These figures are shown in the next slides

4. MATLAB “Deep Belief Network for chest x-rays”—Step 4) DBN live to test 2 Chest x-rays images

The screenshot displays the MATLAB IDE interface. The top menu bar includes HOME, PLOTS, APPS, EDITOR, PUBLISH, and VIEW. The workspace on the left shows variables: edbn (1x1 struct), er (0.1429), image (6), images_x_T (14x784 double), images_x_V (7x784 double), imds (1x1 ImageDatastore), imdsTrain (1x1 ImageDatastore), imdsValidation (1x1 ImageDatastore), label_y_T (14x10 double), label_y_V (7x10 double), n_T (14), and n_V (7). The current folder is 'Chest_X_Rays', containing subfolders 'edbn-master' and 'other', and files 'DBN.m', 'edbnsetup.m', 'edbnset.m', 'edbntrain.m', 'edbntrain.m', 'erbmtrain.m', 'erbmup.m', and 'images.m'. The editor window shows the following code:

```
35 %% Step 4) DBN live to test 2 Chest x-rays images
36 image=2; spike_list = live_edbn(edbn, images_x_V(image, :), opts);
37 output_idx = (spike_list.layers == numel(edbn.sizes));
38 figure; histogram(spike_list.addr(output_idx), 1:edbn.sizes(end));
39 str = sprintf('%d of %d', i, n_V); title(str);
40 xlabel('Category classification'); ylabel('Histogram Spike Count');
41 str = sprintf('Class Classification Spikes test images %d', image); title(str);
42
43 image=6; spike_list = live_edbn(edbn, images_x_V(image, :), opts);
44 output_idx = (spike_list.layers == numel(edbn.sizes));
45 figure; histogram(spike_list.addr(output_idx), 1:edbn.sizes(end));
46 str = sprintf('%d of %d', i, n_V); title(str);
47 xlabel('Category classification'); ylabel('Histogram Spike Count');
48 str = sprintf('Class Classification Spikes test images %d', image); title(str);
```

The command window shows the execution of the code, including histogram generation and classification results:

```
xlabel('Category classification'); ylabel('Histogram Spike Count');
str = sprintf('Class Classification Spikes test images %d', image); title(str);

image=6; spike_list = live_edbn(edbn, images_x_V(image, :), opts);
output_idx = (spike_list.layers == numel(edbn.sizes));
figure; histogram(spike_list.addr(output_idx), 1:edbn.sizes(end));
str = sprintf('%d of %d', i, n_V); title(str);
xlabel('Category classification'); ylabel('Histogram Spike Count');
str = sprintf('Class Classification Spikes test images %d', image); title(str);
Completed 2000 input spikes occurring over 4.00 seconds, in 6.224 seconds of real time.
Completed 2000 input spikes occurring over 4.00 seconds, in 5.660 seconds of real time.
```

Copy and paste in the command prompt: “Step 4) DBN live to test 2 Chest x-rays images”. Where the test images are analyzing showing the dynamic spikes and then its histogram is generated to see the class (category) obtain in the classification. *These figures are shown in the next slides.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

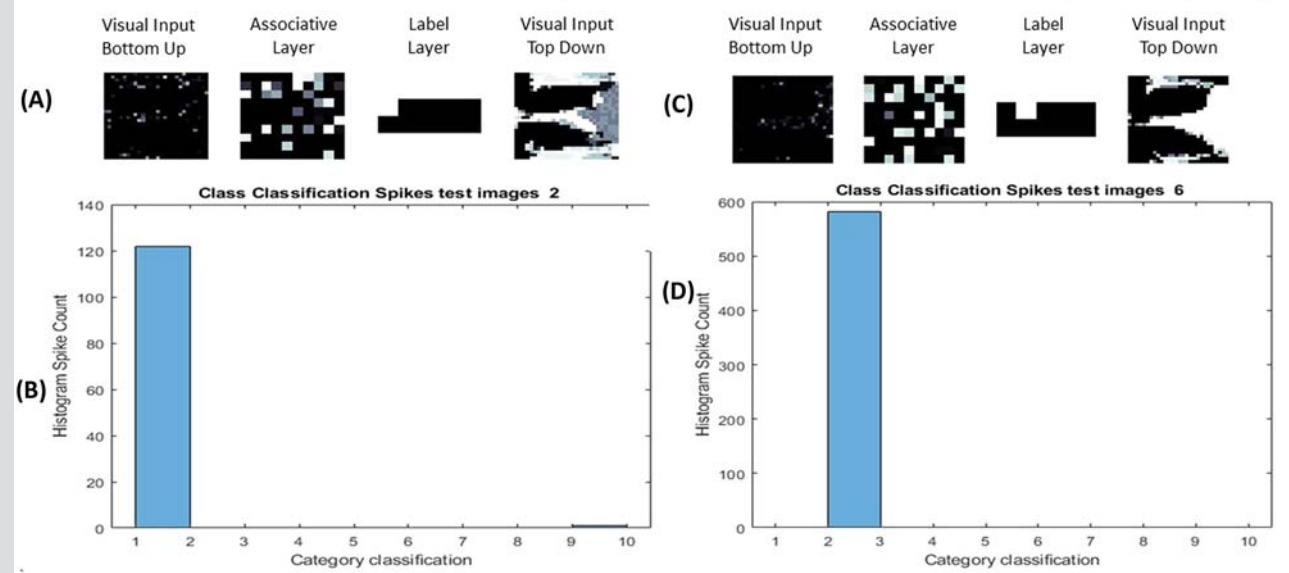
(Continued)

(Continued)

Slide	Description	Screen figure
-------	-------------	---------------

5	<p>MATLAB "Deep Belief Network for chest X-rays" — Step 4) Figures that show the Results DBN live testing 2 Chest X-ray images</p> <p>When Step 4) is run, the image num 2 and 6 are analyzed as shown in A) and C), where the process dynamically compares each image with each generative image during the training. The histogram for image 2 in B) shows that it is classified as "class 1 or Normal Chest X-Rays," and the histogram for image 6 shows that it is classified as "class 2 or abnormal Chest X-rays"</p>	
---	---	--

5. MATLAB "Deep Belief Network for chest x-rays"-Step 4) Figures that show the Results DBN live testing 2 Chest x-rays images



When step 4 is run, the image num 2 and 6 are analyzed as shown in A) and C), where the process dynamically of compare each image with each generative images during the training. The histogram for image 2 in B) shows that is classified as "class 1 or Normal Chest X-Rays", and the histogram for image 6 in D) shows that is classified as "class 2 or abnormal Chest x-rays".

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

The “Deep Belief Network” based on node-neurons from “Spiking Neural Networks (SNN)” applying the “Siegert Neurons mathematical model” allows:

- Generative graphical representation producing all possible values that can be generated, as shown in Table of slides 6.7: slide 3, where 100 images were generated from 14 images used for the training to generate the “Deep Belief Network model.”
- For testing or validation of a series of figures obtained as indicated in the Table of slides 6.7: slide 5. The images are compared during the process at the “Visual Input Bottom-Up,” “Associative Layer,” “Label Layer,” and “Visual Input Top-Down Layer.” And finally, an “histogram” indicates the corresponding “classification category” as “1” for image 2, and “2” for images 6 from the validation dataset.

Recommendation

This example was developed as a simulation for “DBN proof of concept” with only 21 images with low resolution of “28–28 pixels” to be run on all most computers, A higher number of images with higher resolution can be used with a computer that has a parallel computing platform and programming model developed for general computing on “graphical processing units (GPUs),” such as “NVIDIA.”

6.5 Evolutionary Deep Neural Networks types

Evolutionary Deep Neural Networks imply all new ways to automate features for the required engineering and new architecture designs based on observations of how the human brain processes, accesses, analyzes, deduces, and reasons, applying new technologies as they become available. Examples are “Capsule Networks (CapsNet),” “Attention networks (AN),” and others.

6.5.1 Capsule Networks

“Capsule Networks (CapsNet)” was invented by Geoffrey Hinton, one of the godfathers of “Deep Learning.” It is a biology inspired alternative to pooling, where “neurons are related to multiple weights using a numeric vector” instead of just one weight as a scalar value. This method uses a new concept known as “capsules” that perform quite complicated internal computations on their inputs and then “encapsulate” the results of these computations into a small vector of highly informative data including where is the object encapsulated, the scale of the object, its angle and other spatial information, and “dynamic routing between capsules algorithm” that allow capsules to communicate with each other and create representations similar to scene graphs in computer graphics. The “CapsNet” uses a lot of computational resources of “GPU

(Graphics Processing Units) type” and it can be used to obtain a better hierarchical relationship model, to more closely mimic biological neural organization [62]. As a specific definition for “a capsule” is a group of neurons where the activities of the object or part of the object encapsulated are represented as an “activity vector,” that includes the probability that the entity exists and its orientation represents the instantiation parameters.

The “Convolutional Neural Network (CNN)” has a “convolutional layer” to detect important features from the image pixels, to learn to detect simple features, such as edges and color gradients, and in higher layers to combine simple features into more complex features. The main problem with “CNN” is that the use of “max pooling” or “successive convolutional layers” raises the performance, while “max pooling” loses valuable information. This is a big problem when detailed information must be preserved throughout the network, such as in “semantic segmentation.” To resolve this problem some CNN complex architectures are built to recover some of the lost information.

In “CapsNet” the following concepts are applied:

- The traditional neurons that receive “scalar values (x_i)” as input and deliver “scalar values (y_i)” as output were replaced in “CapsNet” with “capsules” that use “vector values (\mathbf{u}_i)” as input and deliver “vector values (\mathbf{v}_j)” as output.
- The “CNN” traditional operations in neurons for: “weighting $a_j = \sum W_i x_i + b$ ” and “nonlinearity activation function $h_{w,b}(x) = f(a_j)$ ” “in “CapsNets” were replaced by: “Affine Transformation $\hat{\mathbf{u}}_{ji} = \mathbf{w}_{ij} \mathbf{u}_i$ ”, “weighting $\mathbf{S}_j = \sum \mathbf{c}_{i,j} \hat{\mathbf{u}}_{ji}$,” and “nonlinearity activation function $\mathbf{v}_j = \frac{||\mathbf{S}_j||^2}{1 + ||\mathbf{S}_j||^2} \frac{\mathbf{S}_j}{||\mathbf{S}_j||}$.”

The “CapsNet” has an architecture that can be divided into two parts, as shown in Fig. 6.21: an “encoder” using supervised learning with a multilabel max norm loss, and a “decoder” using unsupervised learning for reconstruction loss, where:

- “Encoder” is a part of the network that takes images inputs to recognize the images from a predefined group of 9, in this example with “28 × 28” pixel size resolution, and learns to encode it onto a 10 × 16-dimensional vector, as shown in the upper part of Fig. 6.21 using the following three layers:
 - “Convolutional layer” is used to detect basic features in the 2D image. In the “CapsNet,” the “convolutional layer” has “256 kernels” with size of “9 × 9 × 1” and stride “1*,” followed by “ReLU activation,” that performs a threshold operation to each element of the input, where any value less

Evolutionary Deep Neural Networks: Capsule Network (CapsNet)

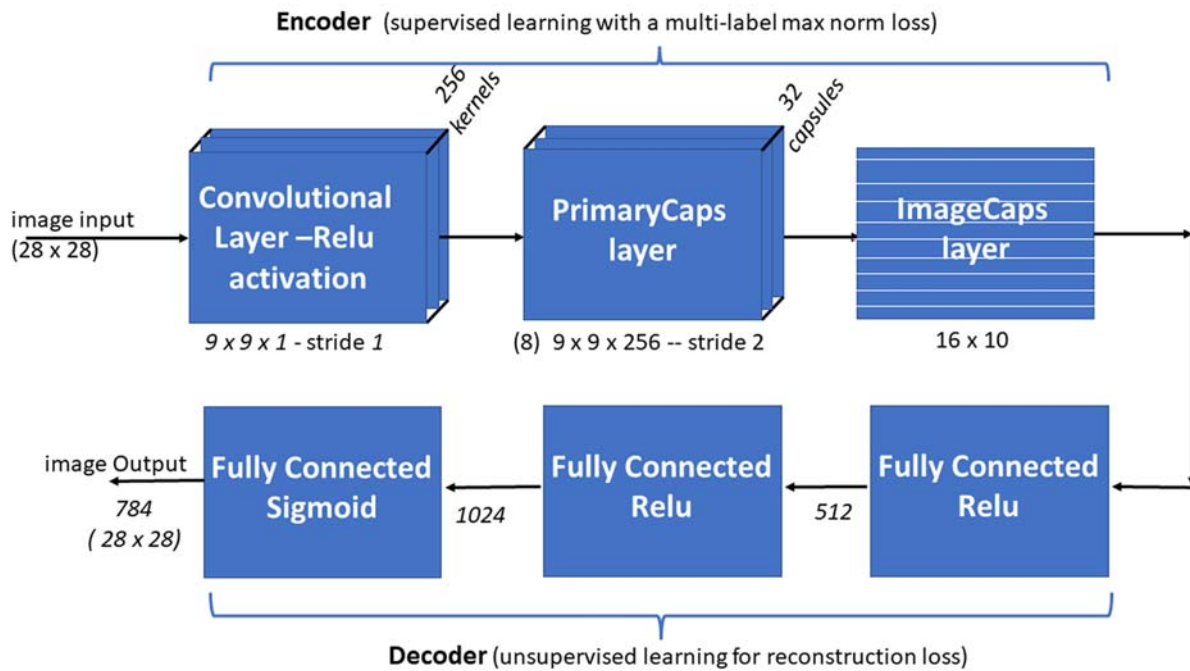


FIGURE 6.21 Capsule Network layer structure: Encoder and Decoder.

than zero is set to zero. Each kernel has 1 bias, thus this layer has $(9 \times 9 + 1) \times 256 = 20,992$ trainable parameters that need to be trained.

Note: Stride is a parameter of the neural network's filter that modifies the amount of movement over the image or video. For example, if a neural network's stride is set to 1, the filter will move one pixel at a time.

- "PrimaryCaps layer" takes the basic features detected by the "convolutional layer" and creates combination of the features. It has 32 "primary capsules," where each capsule has eight " $9 \times 9 \times 256$ " convolutional kernels with stride "2." All parameters need also to be trained.
- "ImageCaps layer" has 1 digit for category, for example, the MINST dataset has 10 capsules, one for each digit from 0 to 9 in a matrix of " 16×10 " to be recognized, this layer is also known as "DigitCaps."
- "Decoder," as shown in the lower part of Fig. 6.21, takes a 16-dimensional vector from the "encoder" and learns to decode it into an image onto a 28×28 image with the loss function based on "Euclidian distance" between the reconstructed image and the input image. For this, the "Decoder" also uses the following three layers:
 - "Fully Connected layer #1" receives as input a " 16×10 " matrix that are all directed to each of the "512" neurons in the output of this layer.

- "Fully Connected layer #2" receives as input "512" values that are all directed to each of the "1024" neurons in the output of this layer.
- "Fully Connected layer #3" receives as input "1024" values that are all directed to each of the "784" neurons in the output of this layer. This is reshaping back to the original size of " 28×28 " pixels image.

Note*: "CapsNet" has a state-of-the-art performance on simple datasets, but in a large dataset the excess amount of information in images throws off the capsules. They are continuously researching for new applications by applying different technologies to handle the images information.

Some examples of "Capsule Networks (CapsNet)" applied to "Biomedical Engineering" are:

- "Multichannel EEG-based emotion recognition via a multilevel features guided capsule network" by Liu et al. [63]. In this paper, they propose an effective "multilevel features guided capsule network (MLF-CapsNet)" for "multichannel EEG-based emotion recognition." The "MLF-CapsNet" is an end-to-end framework, which can simultaneously extract features from the raw "electroencephalogram (EEG)" signals and determine the emotional states. Compared with the original "CapsNet," it incorporates multilevel feature maps learned by different layers in forming the

primary capsules so that the capability of feature representation can be enhanced. In addition, it uses a bottleneck layer to reduce the number of parameters and accelerate the speed of calculation.

- “Convolutional capsnet: A novel artificial neural network approach to detect COVID-19 disease from X-ray images using capsule networks” by Toramant et al. [64]. In this study, a novel artificial neural network, “Convolutional CapsNet” for the detection of COVID-19 disease is proposed by using “chest X-ray images with capsule networks.” The proposed approach is designed to provide fast and accurate diagnostics for COVID-19 diseases with binary classification (COVID-19, and No-Findings), and multiclass classification (COVID-19, No-Findings, and Pneumonia).
- “Tag recommendation by text classification with attention-based capsule network” by Lei et al. [65]. Tag recommendation has been attracting much attention with the growth of digital resources. The goal of a tag recommendation system is to provide a set of tags for a piece of text to ease the tagging process done manually by a user. These tags have been shown to enhance the capabilities of search engines for navigating, organizing, and searching content. However, tagging text manually is time-consuming and labor-intensive. In this paper, they introduce a “tag recommendation by text classification.” They explore the

“capsule network with dynamic routing for the tag recommendation task.” The “capsule network” encodes the intrinsic spatial relationship between a part and a whole constituting viewpoint invariant knowledge that automatically generalizes to novel viewpoints. In addition, an attention mechanism is incorporated into the capsule network to distill important information from the input documents.

The “CapsNet” uses a lot of computational resources of “GPU (Graphics Processing Units) type” and it can be used to obtain a better model hierarchical relationship, to more closely mimic biological neural organization [62]. As a specific definition for “a capsule” is a group of neurons where the activities of the object or part of the object encapsulated are represented as an “activity vector,” that includes the probability that the entity exists and its orientation to represent the instantiation parameters.

6.5.2 Attention networks

“Attention network (AN)” as shown in Fig. 6.22A, uses frequently a “Recurrent Neural Network RNN (v_i),” and an “Attention Mechanism (h_i, α_i)” to combat information decay by separately storing previous network states, and switching attention between the states to obtain an output (y_i). Where in the “RNN are encoders and decoders”

Evolutionary Deep Neural Networks: Attention Network (AN)

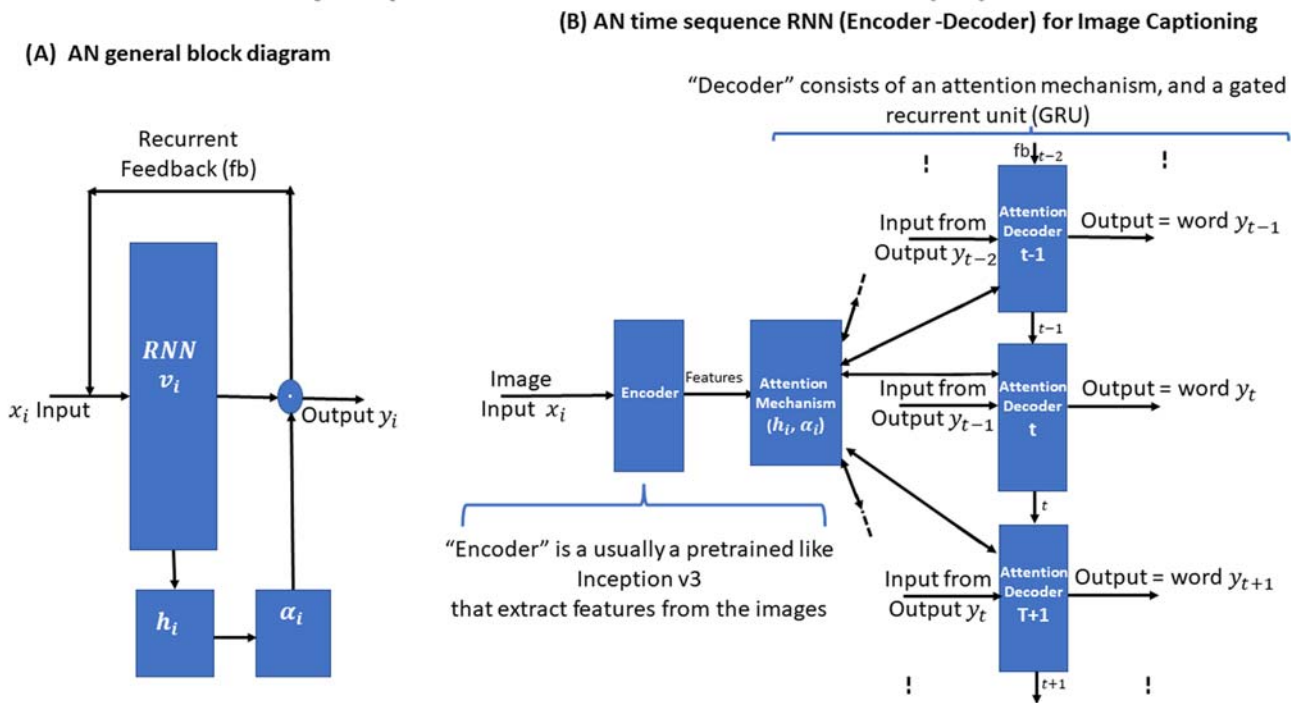


FIGURE 6.22 Attention Network: (A) A general block diagram and (B) A time sequence RNN (Encoder–Decoder) for Image Captioning.

where the hidden states of each iteration in the encoding layers are stored in memory cells (h_i) as shown in Fig. 6.22B, where the “decoding layers” are connected to the “encoding layers,” but it also receives data from the memory cells filtered by an attention context (α_i). This filtering step adds context for the decoding layers stressing the importance of some particular features.

The “attention network” producing this context is trained using the error signal from the output of decoding layer. Moreover, the attention context can be visualized giving valuable insight into which input features correspond with what output features. “AN” also includes the transformer architecture [66], where transformers use attention mechanisms to gather information about the relevant context of a given word, and then encode that context in the vector that represents the word. So, in a sense, attention and transformers are about smarter representations. “AN” typically has the objective of accomplish text classification based in “Words make sentences and sentences make documents,” It uses “stacked recurrent neural networks” on word level followed by “attention model” to extract such words that are important to the meaning of the sentence and aggregate the representation of those informative words to form a sentence vector. Then the same procedure applied to the derived sentence vectors which then generate a vector who conceives the meaning of the given document and that vector can be passed further for text classification* [67].

Some examples of “Attention networks (AN)” applied to “Biomedical Engineering” are:

- “SiamAtt: Siamese attention network for visual tracking” by Yang et al. [68]. Visual attention has recently achieved great success and wide application in deep neural networks. They introduce an “attention branch in the region proposal network” that contains a classification branch and a regression branch. They perform foreground–background classification by combining the scores of the classification branch and the attention branch. The regression branch predicts the bounding boxes of the candidate regions based on the classification results.
- “Domain-specific Involvement of the Right Posterior Parietal Cortex in Attention Network and Attentional Control of ADHD: A Randomized, Cross-over, Sham-controlled tDCS Study” by Salehinejad et al. [69]. “Transcranial direct current stimulation (tDCS)” has been increasingly used in “attention-deficit hyperactivity disorder (ADHD)” with mixed results. In this paper they explored the effects of “anodal tDCS” over the “right posterior parietal cortex (r-PPC)” on “attentional functioning (i.e., attention networks, selective attention, shifting attention)” and response inhibition in “ADHD children.”
- “Cross-modal recipe retrieval via parallel- and cross-attention networks learning” by Cao et al. [70].

“Cross-modal recipe retrieval” refers to the problem of retrieving an image from a list of image candidates given a textual recipe as the query, or the reverse side. They study the problem of “cross-modal recipe retrieval from the viewpoint of parallel- and cross-attention networks learning.” Specifically, they exploit a “parallel-attention network” to independently learn the “attention weights of components in images and recipes.” Thereafter, a “cross-attention network is proposed to explicitly learn the interplay between images and recipes,” which simultaneously considers “word-guided image attention and image-guided word attention.” Lastly, the learned representations of images and recipes stemming from “parallel- and cross-attention networks are elaborately connected and optimized using a pairwise ranking loss.”

Note*: Please see an “AN” example at Chapter 7, Research 7.2 “Attention network using Long/Short-Term Memory to Classify Text of COVID-19 symptoms”.

References

- [1] M. Deng, T. Meng, J. Cao, S. Wang, J. Zhang, H. Fan, Heart sound classification based on improved MFCC features and convolutional recurrent neural networks, *Neural Netw.* Volume 130 (2020) 22–32. Available from: <https://doi.org/10.1016/j.neunet.2020.06.015>. ISSN 0893-6080.
- [2] E. Messner, M. Fediuk, P. Swatek, S. Scheidl, F.-M. Smolle-Jüttner, H. Olschewski, et al., Multi-channel lung sound classification with convolutional recurrent neural networks, *Computers Biol. Med.* Volume 122 (2020) 103831. Available from: <https://doi.org/10.1016/j.compbiomed.2020.103831>. <http://www.sciencedirect.com/science/article/pii/S0010482520301955>. ISSN 0010-4825.
- [3] R. Casal, L.E. Di Persia, G. Schlotthauer, Classifying sleep–wake stages through recurrent neural networks using pulse oximetry signals, *Biomed. Signal Process. Control* Volume 63 (2021) 102195. Available from: <https://doi.org/10.1016/j.bspc.2020.102195>. <http://www.sciencedirect.com/science/article/pii/S1746809420303311>. ISSN 1746-8094.
- [4] Available from: <https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks> (accessed 09.06.20).
- [5] A. Ghosh, Recurrent neural network and long short-term memory. Available from: <https://medium.com/datadriveninvestor/recurrent-neural-networks-and-long-short-term-memory-5d17bdbdfc00>, October 2018 (accessed 24.07.20).
- [6] J. Cai, J. Hu, X. Tang, T.-Y. Hung, Y.-P. Tan, Deep historical long short-term memory network for action recognition, *Neurocomputing* Volume 407 (2020) 428–438. Available from: <https://doi.org/10.1016/j.neucom.2020.03.111>. <http://www.sciencedirect.com/science/article/pii/S0925231220306068>. ISSN 0925-2312.
- [7] Y. Chen, J. Lv, Y. Sun, B. Jia, Heart sound segmentation via Duration Long–Short Term Memory neural network, *Appl. Soft Comput.* Volume 95 (2020) 106540. Available from: <https://doi.org/10.1016/j.asoc.2020.106540>. ISSN 1568-4946.

- [8] W. Zhang, J. Han, S. Deng, Abnormal heart sound detection using temporal quasi-periodic features and long short-term memory without segmentation, *Biomed. Signal Process. Control* Volume 53 (2019) 101560. Available from: <https://doi.org/10.1016/j.bspc.2019.101560>. ISSN 1746-8094.
- [9] Available from: https://en.wikipedia.org/wiki/Long_short-term_memory (accessed 24.07.19).
- [10] F. Shahid, A. Zameer, M. Muneeb, Predictions for COVID-19 with deep learning models of LSTM, GRU and Bi-LSTM, *Chaos Soliton. Fract.* Volume 140 (2020) 110212. Available from: <https://doi.org/10.1016/j.chaos.2020.110212>. ISSN 0960-0779.
- [11] T. Shi, S. Huang, L. Chen, Y. Heng, Z. Kuang, L. Xu, et al., A molecular generative model of ADAM10 inhibitors by using GRU-based deep neural network and transfer learning, *Chemom. Intell. Lab. Syst.* Volume 205 (2020) 104122. Available from: <https://doi.org/10.1016/j.chemolab.2020.104122>. ISSN 0169-7439.
- [12] Available from: <https://medalhelp.org/falls-in-the-elderly/> (accessed 23.06.20).
- [13] C.J. Sporer, P. McClure, N. Kriegeskorte, *Front. Psychol.*, September 12, 2017 | <https://doi.org/10.3389/fpsyg.2017.01551> (accessed 12.08.19).
- [14] Recurrent Convolutional Neural Networks: a better model of biological object recognition. Available from: <https://www.frontiersin.org/articles/10.3389/fpsyg.2017.01551/full> (accessed 24.06.20).
- [15] B. Abraham, M.S. Nair, Computer-aided detection of COVID-19 from X-ray images using multi-CNN and Bayesnet classifier, *Biocybern. Biomed. Eng.* (2020). Available from: <https://doi.org/10.1016/j.bbe.2020.08.005>. ISSN 0208-5216.
- [16] Available from: <https://www.mathworks.com/help/vision/examples/train-an-object-detector-using-you-only-look-once.html> (accessed 09.05.20).
- [17] Available from: <https://www.mathworks.com/help/vision/ug/getting-started-with-r-cnn-fast-r-cnn-and-faster-r-cnn.html> (accessed 09.05.20).
- [18] L. Cantini, M. Caselle, Hope4Genes: a Hopfield-like class prediction algorithm for transcriptomic data, *Sci. Rep.* 9 (2019) 337.
- [19] Available from: <https://www.mayoclinic.org/tests-procedures/chest-x-rays/about/pac-20393494> (accessed 14.07.20).
- [20] L.A. Passos, L.A. de Souza Jr., R. Mendel, A. Ebigbo, A. Probst, H. Messmann, et al., Barrett's esophagus analysis using infinity Restricted Boltzmann Machines, *J. Vis. Commun. Image Represent.* Volume 59 (2019) 475–485. Available from: <https://doi.org/10.1016/j.jvcir.2019.01.043>. ISSN 1047-3203.
- [21] B. Xiaojun, W. Haibo, Contractive slab and spike convolutional deep Boltzmann Machine, *neurocomputing* Volume 290 (2018) 208–228. Available from: <https://doi.org/10.1016/j.neucom.2018.02.048>. ISSN 0925-2312.
- [22] O.A. Zoubi, M. Awad, N.K. Kasabov, Anytime multipurpose emotion recognition from EEG data using a Liquid State Machine based framework, *Artif. Intell. Med.* Volume 86 (2018) 1–8. Available from: <https://doi.org/10.1016/j.artmed.2018.01.001>. ISSN 0933-3657.
- [23] Y. Jin, P. Li, Performance and robustness of bio-inspired digital liquid state machines: a case study of speech recognition, *Neurocomputing* Volume 226 (2017) 145–160. Available from: <https://doi.org/10.1016/j.neucom.2016.11.045>. ISSN 0925-2312.
- [24] Y. Zhou, Y. Jin, J. Ding, Surrogate-assisted evolutionary search of spiking neural architectures in liquid state machines, *Neurocomputing* Volume 406 (2020) 12–23. Available from: <https://doi.org/10.1016/j.neucom.2020.04.079>. ISSN 0925-2312.
- [25] H. Hazan, L.M. Manevit, Topological constraints and robustness in liquid state machines, *Expert. Syst. Appl.* 39 (2) (2012) 1597–1606. Available from: <https://doi.org/10.1016/j.eswa.2011.06.052>.
- [26] J. Herbert, H. Haas, Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless communication, *Science* 304 (5667) (2004) 78–80.
- [27] H.-H. Kim, J. Jeong, Decoding electroencephalographic signals for direction in brain-computer interface using echo state network and Gaussian readouts, *Comput. Biol. Med.* Volume 110 (2019) 254–264. Available from: <https://doi.org/10.1016/j.compbiomed.2019.05.024>. ISSN 0010-4825.
- [28] V.H.A. Ribeiro, G. Reynoso-Meza, H.V. Siqueira, Multi-objective ensembles of echo state networks and extreme learning machines for streamflow series forecasting, *Eng. Appl. Artif. Intell.* Volume 95 (2020) 103910. Available from: <https://doi.org/10.1016/j.engappai.2020.103910>. ISSN 0952-1976.
- [29] C. Lu, P. Xu, L.-h Cong, Fault diagnosis model based on granular computing and echo state network, *Eng. Appl. Artif. Intell.* Volume 94 (2020) 103694. Available from: <https://doi.org/10.1016/j.engappai.2020.103694>. ISSN 0952-1976.
- [30] Available from: <https://www.izhikevich.org/publications/spikes.htm> (accessed 12.07.20).
- [31] Available from: <https://www.radiologyinfo.org/en/info.cfm?pg=pneumonia> (accessed 14.07.20).
- [32] Available from: <https://litfl.com/imaging-of-covid-19-pneumonia-a-critical-care-perspective/> (accessed 14.07.20).
- [33] W.-j. Guan, Z.-y. Ni, et al. Clinical characteristics of Coronavirus Disease 2019 in China, Wei-jie Guan, PhD. Available from: <https://www.nejm.org/doi/10.1056/NEJMoa2002032>.
- [34] H.Y.F. Wong, H.Y.S. Lam, et. al. Frequency and distribution of chest radiographic findings in COVID-19 positive patients, 2020. Available from: <https://doi.org/10.1148/radiol.2020201160>.
- [35] K. Teuvo, Self-organized formation of topologically correct feature maps, *Biol. Cybern.* 43 (1) (1982) 59–69.
- [36] P. Melin, J.C. Monica, D. Sanchez, O. Castillo, Analysis of spatial spread relationships of Coronavirus (COVID-19) pandemic in the world using self organizing maps, *Chaos Soliton. Fract.* Volume 138 (2020) 109917. Available from: <https://doi.org/10.1016/j.chaos.2020.109917>. ISSN 0960-0779.
- [37] Q. Yang, G.-L. Tian, J.-W. Qin, B.-Q. Wu, L. Tan, L. Xu, et al., Coupling bootstrap with synergy self-organizing map-based orthogonal partial least squares discriminant analysis: atable metabolic biomarker selection for inherited metabolic diseases, *Talanta* Volume 219 (2020) 121370. Available from: <https://doi.org/10.1016/j.talanta.2020.121370>. ISSN 0039-9140.
- [38] T. Kunz, L. Rieber, S. Mahony, Assessing relationships between chromatin interactions and regulatory genomic activities using the self-organizing map, *Methods* (2020). Available from: <https://doi.org/10.1016/j.ymeth.2020.07.002>. ISSN 1046-2023.
- [39] Available from: <https://towardsdatascience.com/hands-on-memory-augmented-neural-networks-implementation-part-one-a6a4a88beba3>.
- [40] A. Graves, G. Wayne, I. Danihelka, *Neural Turing Machines*. Available from: <http://www.robots.ox.ac.uk/~tvj/publications/talks/NeuralTuringMachines.pdf> (accessed 26.07.20).
- [41] A. Graves, G. Wayne, I. Danihelka, *Neural Turing Machines*, Google DeepMind, London, UK, 2014. Available from: <https://arxiv.org/pdf/1410.5401v2.pdf> (accessed 07.09.20).

- [42] S.M. Faradonbeh, F. Safi-Esfahani, A review on Neural Turing Machine, Faculty of Computer Engineering, Najafabad Branch, Islamic Azad University, Najafabad, Iran. Big Data Research Center, Najafabad Branch, Islamic Azad University, Najafabad, Iran. Available from: <https://arxiv.org/ftp/arxiv/papers/1904/1904.05061.pdf> (accessed 07.09.20).
- [43] R.B. Greve, E.J. Jacobsen, S. Risi, Evolving neural Turing Machines for reward-based learning, IT University of Copenhagen, Copenhagen, Denmark, 2016. Available from: http://sebastianrisi.com/wp-content/uploads/greve_gecco16.pdf (accessed 07.09.20).
- [44] A. Graves, G. Wayne, M. Reynolds, et al., Hybrid computing using a neural network with dynamic external memory, *Nature* 538 (2016) 471–476. Available from: <https://doi.org/10.1038/nature20101>.
- [45] Available from: <https://deeppmind.com/blog/differentiable-neural-computers/> (accessed 27.07.19).
- [46] Available from: <http://www.asimovinstitute.org/neural-network-zoo/> (accessed 27.07.19).
- [47] R. Sharma, A. Kumar, D. Meena, S. Pushp, Employing differentiable neural computers for image captioning and neural machine translation, *Procedia Computer Sci.* Volume 173 (2020) 2234–2244. Available from: <https://doi.org/10.1016/j.procs.2020.06.028>. ISSN 1877-0509.
- [48] M.S. Rasekh, F. Safi-Esfahani, EDNC: evolving differentiable neural computers, *Neurocomputing* Volume 412 (2020) 514–542. Available from: <https://doi.org/10.1016/j.neucom.2020.06.018>. ISSN 0925-2312.
- [49] Available from: <https://www.who.int/news-room/commentaries/detail/transmission-of-sars-cov-2-implications-for-infection-prevention-precautions> (accessed 25.07.20).
- [50] L.D. Stetzenbach, M.P. Buttner, P. Cruz, Detection and enumeration of airborne biocontaminants, *Curr. Opin. Biotechnol.* 15 (3) (2004) 170–174.
- [51] Available from: <https://globalbiodefense.com/2020/06/24/ucdavis-researchers-help-identify-paths-to-possible-mother-to-child-coronavirus-transmission/> (accessed 25.07.20).
- [52] Available from: <https://www.cebm.net/covid-19/what-is-the-evidence-to-support-the-2-metre-social-distancing-rule-to-reduce-small-droplet-duce-covid-19-transmission/> (accessed 25.07.20).
- [53] Available from: <https://www.hopkinsmedicine.org/health/conditions-and-diseases/vital-signs-body-temperature-pulse-rate-respiration-rate-blood-pressure#:~:text=Normal%20respiration%20rates%20for%20an,to%2016%20breaths%20per%20minute> (accessed 25.07.20).
- [54] W.M. Thurlbeck, Internal surface area and other measurements in emphysema, *Thorax* 22 (6) (1967) 483–496. Available from: <https://doi.org/10.1136/thx.22.6.483>. PMC 471691, PMID 5624577, 07/20/25.
- [55] Available from: <https://www.nytimes.com/2020/04/22/us/politics/social-distancing-coronavirus.html> (accessed 25.08.20).
- [56] Available from: <https://khn.org/news/scientists-want-to-know-more-about-using-uv-light-to-fight-covid-19-spread/> (accessed 25.08.20).
- [57] Black, P.E. “Greedy algorithm.” *Dictionary of Algorithms and Data Structures*, 2005. United States National Institute of Standards and Technology (NIST). Retrieved 17.08.12.
- [58] D.J. Jagannath, D. Raveena Judie Dolly, J. Dinesh Peter, Composite Deep Belief Network approach for enhanced antepartum foetal electrocardiogram signal, *Cognit. Syst. Res.* Volume 59 (2020) 198–203. Available from: <https://doi.org/10.1016/j.cogsys.2019.09.027>. ISSN 1389-0417.
- [59] Z. Lv, L. Qiao, Deep belief network and linear perceptron based cognitive computing for collaborative robots, *Appl. Soft Comput.* Volume 92 (2020) 106300. Available from: <https://doi.org/10.1016/j.asoc.2020.106300>. ISSN 1568-4946.
- [60] N. Qiang, Q. Dong, W. Zhang, B. Ge, F. Ge, H. Liang, et al., Modeling task-based fMRI data via deep belief network with neural architecture search, *Comput. Med. Imaging Graph.* Volume 83 (2020) 101747. Available from: <https://doi.org/10.1016/j.compmedimag.2020.101747>. ISSN 0895-6111.
- [61] S. Ostojic, N. Brunel, From spiking neuron models to linear-nonlinear models, *PLoS Comput. Biol.* 7 (1) (2011). Available from: <https://pdfs.semanticscholar.org/4720/fea0096dad1ea6eaf5b8329c897476255abd.pdf> (accessed 03.08.20).
- [62] Sabour S., Frosst N., Hinton, G.E. Dynamic Routing Between Capsules (2017).
- [63] Y. Liu, Y. Ding, C. Li, J. Cheng, R. Song, F. Wan, et al., Multi-channel EEG-based emotion recognition via a multi-level features guided capsule network, *Comput. Biol. Med.* Volume 123 (2020) 103927. Available from: <https://doi.org/10.1016/j.combiomed.2020.103927>. ISSN 0010-4825.
- [64] S. Toraman, T.B. Alakus, I. Turkoglu, Convolutional capsnet: a novel artificial neural network approach to detect COVID-19 disease from X-ray images using capsule networks, *Chaos Solit. Fract.* Volume 140 (2020) 110122. Available from: <https://doi.org/10.1016/j.chaos.2020.110122>. ISSN 0960-0779.
- [65] K. Lei, Q. Fu, M. Yang, Y. Liang, Tag recommendation by text classification with attention-based capsule network, *Neurocomputing* Volume 391 (2020) 65–73. Available from: <https://doi.org/10.1016/j.neucom.2020.01.091>. ISSN 0925-2312.
- [66] Available from: <http://www.asimovinstitute.org/neural-network-zoo/> (accessed 30.07.19).
- [67] H. Sankesara, Hierarchical Attention Networks, August 2018. Available from: <https://medium.com/analytics-vidhya/hierarchical-attention-networks-d220318cf87e> (accessed 30.07.19).
- [68] K. Yang, Z. He, Z. Zhou, N. Fan, SiamAtt: Siamese attention network for visual tracking, *Knowl. Syst.* Volume 203 (2020) 106079. Available from: <https://doi.org/10.1016/j.knsys.2020.106079>. ISSN 0950-7051.
- [69] M.A. Salehinejad, E. Ghayerin, V. Nejati, F. Yavari, M.A. Nitsche, Domain-specific involvement of the right posterior parietal cortex in attention network and attentional control of ADHD: a randomized, cross-over, sham-controlled tDCS study, *Neuroscience* Volume 444 (2020) 149–159. Available from: <https://doi.org/10.1016/j.neuroscience.2020.07.037>. ISSN 0306-4522.
- [70] D. Cao, J. Chu, N. Zhu, L. Nie, Cross-modal recipe retrieval via parallel- and cross-attention networks learning, *Knowl. Syst.* Volume 193 (2020) 105428. Available from: <https://doi.org/10.1016/j.knsys.2019.105428>. ISSN 0950-7051.
- [71] Lianli Gao, Xuanhan Wang, Jingkuan Song, Yang Liu, Fused GRU with semantic-temporal attention for video captioning, *Neurocomputing*, 395 (0925-2312) (2020) 222–228. Available from: <https://doi.org/10.1016/j.neucom.2018.06.096>.

- [72] Zhiyong Liu, Chuan Yang, Jun Huang, Shaopeng Liu, Yumin Zhuo, Xu Lu, Deep learning framework based on integration of S-Mask R-CNN and Inception-v3 for ultrasound image-aided diagnosis of prostate cancer, *Future Generation Computer Systems* 114 (0167-739X) (2021) 358–367. Available from: <https://doi.org/10.1016/j.future.2020.08.015>.
- [73] M. Frei, F.E. Kruis, FibeR-CNN: Expanding Mask R-CNN to improve image-based fiber analysis, *Powder Technology* 377 (0032-5910) (2021) 974–991. Available from: <https://doi.org/10.1016/j.powtec.2020.08.034>.
- [74] Miguel Atencia, Gonzalo Joya, Francisco Sandoval, Identification of noisy dynamical systems with parameter estimation based on Hopfield neural networks, *Neurocomputing* 121 (0925-2312) (2013) 14–24. Available from: <https://doi.org/10.1016/j.neucom.2013.01.030>.
- [75] Xin Huang, Kuangrong Hao, Yongsheng Ding, Human fringe skeleton extraction by an improved Hopfield neural network with direction features, *Neurocomputing*, 87 (0925-2312) (2012) 99–110. Available from: <https://doi.org/10.1016/j.neucom.2012.02.010>.
- [76] J. Vrábel, P. Pořízka, J. Kaiser, Restricted Boltzmann Machine method for dimensionality reduction of large spectroscopic data, *Spectrochim. Acta Part B At. Spectrosc.* Volume 167 (2020) 105849. Available from: <https://doi.org/10.1016/j.sab.2020.105849>. ISSN 0584-8547.
- [77] Available from: <https://github.com/Andrea-V/Restricted-Boltzmann-Machine>, 1, 2018 (accessed 10.10.21).

Further reading

Available from: <https://www.osti.gov/servlets/purl/1405258> (accessed 25.07.19).

Available from: <https://pjreddie.com/darknet/yolo/> (accessed 05.09.20).

This page intentionally left blank

Cognitive learning and reasoning models applied to biomedical engineering

7.1 Introduction

In this chapter we will focus on many “*prestudies and preanalysis of different Biomedical Engineering problems that need to be developed with specialized research projects applying Cognitive Learning and Reasoning (CL&R) algorithm, that can be integrated to the Proposed General Architecture framework of a Cognitive Computing Agents System (AI-CCAS)*” with special emphasis on “*Cognitive Learning and its relationship with the neuroscience of reasoning, proposed as Cognitive Learning Reasoning (CL&R) using Cognitive Computing (CC)*.” Through this book, we have studied many interactions of different “*human illnesses, diseases, and disorders*,” where “*human illness*” is defined as body damage that needs to be cured, such as infections, injuries, cells degeneration, etc.; “*human diseases*” are defined as states or reactions that must be managed, such as pain, discomfort, weakness, fatigue, etc.; and “*human disorders*” are defined as functions or abnormalities that must be treated, such as physical, mental, genetic, emotional/behavioral, and functional issues. The complexity needed for the analysis is boundless, and can only be analyzed through multidisciplinary science, as indicated in Chapter 1, Biomedical Engineering and the Evolution of Artificial Intelligence, Fig. 1.1, via the interaction of fields such as “*Biomedical Engineering*,” “*Neurology*,” “*Cognitive Sciences*,” and “*Computer Science*” using tools with “*exponential technologies*” such as “*Artificial intelligence (AI)*” and others through “*continuous exponential evolution*,” as shown in Chapter 1, Fig. 1.3, that are, “*Machine Learning (ML)*,” “*Deep Learning (DL)*,” and “*Cognitive Computing (CC)*.” The main purpose is to obtain useful “*AI models*” that can help to analyze human health problems. Now it is the time to apply them and many others in research projects.

Note: Each “AI-CCAS” step and its iteration with “CL&R” using “CC” is explained with more detail in the next sections.

7.2 Artificial intelligence and Cognitive Computing Agents System (AI-CCAS)

The “*main objective of AI-CCAS is to create specific AI cognitive models based on special human cognitive functionality factors of the affected patients*,” in order to detect, evaluate, and classify different important human factors, such as “*Gesture recognition*,” “*Speech generation*,” “*Mood behavior*,” “*Expressed Sentiments*,” and others to obtain a real feedback of “*nonmotor neurological symptoms*,” that until now in general ways are evaluated based on procedures, questions, and answers of patients, caregivers, and others. Developing and using a framework, such as “*Cognitive Computing Agents System (AI-CCAS)*,” medical doctors and specialists can take better decisions based on improving feedback with parameters (values) using real measurement of behavioral information variables, that can be stored, and then can be available as “*AI-medical history*,” that is, “*AI Information retrieval models*,” “*AI Knowledge formation*,” “*AI extraction methods*,” and many other new techniques that are necessary for development. “*AI specialized tools evaluate human behaviors*” accordingly, and can improve the “*follow-up of the patient’s diseases progression*” by evaluating if their treatments, medicines, health exercises, etc. are well prescribed in amount or effects, obtaining in each patient visit measurements to be compared with expected results to determine if they need to be readjusted.

Different methods and techniques with examples have been explained and developed in many research examples through this book for the components of the proposed framework. As shown in Fig. 7.1, “*Cognitive Computing Agents System (AI-CCAS)*” are: “*speech recognition*,” “*visual recognition*,” “*biomedical instruments (bioinstruments)*,” “*AI-ML-DL-CC dataset obtainment for biomedical engineering including cognitive detection human-like abilities*,” “*AI technologies evolutions as cognitive computing*,” “*AI-ML-DL-CC models obtainment including cognitive models*,” “*AI-ML-DL-CC human functionalities analysis including cognitive evaluations*,” “*inference*”

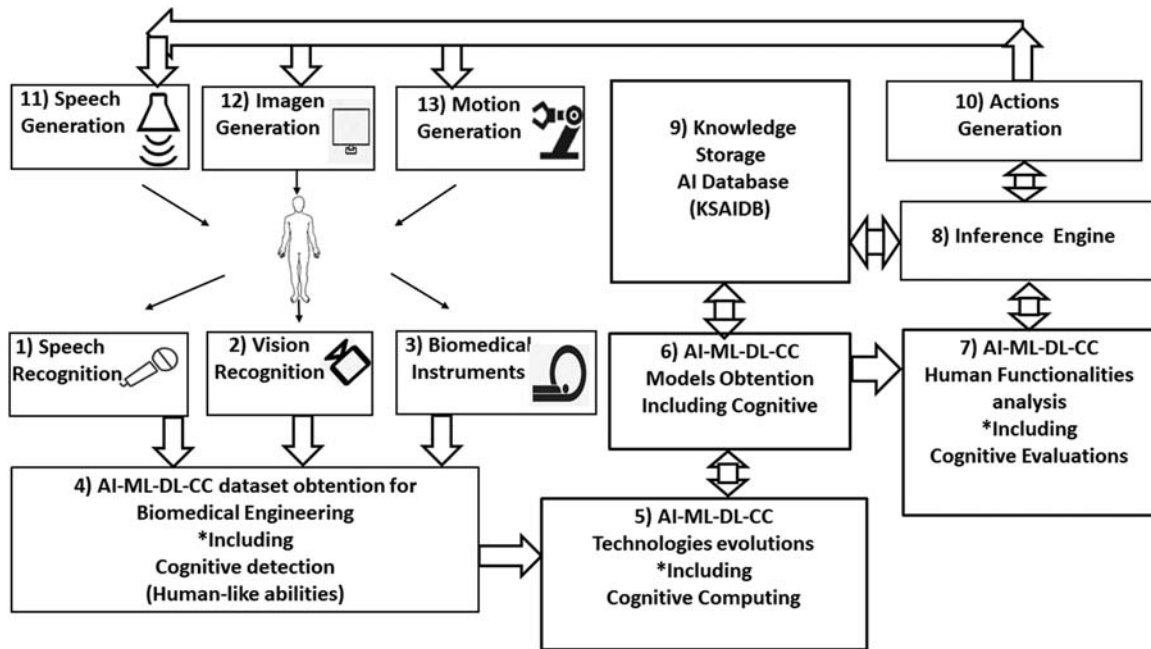


FIGURE 7.1 Cognitive Computing Agents System (AI-CCAS) framework showing its main components.

engine,” “actions generation,” “speech generation,” “imagen generation,” and “motion generation.” The descriptions of each component shown at Fig. 7.1 can be summarized as:

1. “Speech recognition” is frequently used in “Natural Language Processing (NLP)” in two different ways: “Natural Language Understanding” and “Natural Language Generation” to facilitate the obtainment of different “AI models.”
2. “Vision recognition” is used to perform a large number of “machine learning” and “deep learning” applications based on visual tasks through “computer vision,” “optical character recognition,” “labeling the content of images with meta-tags,” “performing image content search,” and other “AI applications,” such as guiding autonomous robots, self-driving cars, and accident avoidance systems.
3. “Biomedical instruments (bioinstruments)” are frequently used for data acquisition as a specialized instrument, with appropriate sensors over the biological body to record, store, measure, and display information about the body’s functions. Examples of bioinstruments that can be used are: “Electrocardiograph (ECG),” a device that measures electrical activity of the heart; “Electromyography (EMG),” a device that measures electrical activities from the muscles; “Electroencephalography (EEG),” a device that measures electric activities from the brain; “Medical image scanners” such as “X-ray,” “Computed tomography (CT),” “Magnetic Resonance Imaging (MRI),”

“Functional Magnetic Resonance Imaging (fMRI),” and “Ultrasound imaging,” and many others [1].

4. “AI-ML-DL-CC dataset obtainment for Biomedical Engineering including Cognitive detection Human-like abilities.*” These datasets are files of information captured through different input sensors and instruments via many methods, as explained in the “speech recognition,” “visual recognition,” “biomedical instruments (bioinstruments),” plus many other datasets that can be found in many places, such as “ImageNet” [2] which contains over 14 million images in over 20,000 categories; “Open Images by Google” [3] that has over 9 million images that map to 6000 categories; “YouTube-8M” [4] a massive dataset taken from YouTube videos with complete annotations on a frame-by-frame basis, “COCO” [5] a large-scale object detection, segmentation, and captioning dataset; and “COIL100” [6] from Columbia University Image Library, which contains 100 images that are mapped from all possible angles. This can be useful for many DL tasks where parts of an image might be occluded and need to be reconstructed. There are many other places that store millions of terabytes of video, email, photos, Facebook posts, and everything else.

Note*: To create a “cognitive human model” it is necessary to include in its design the “cognitive human-like abilities,” as explained in Chapter 1, Section 1.5.19: “Perception,” “Attention,” “Learning,” “Memory,” “Reasoning,” “Actions,” “Emotions,”

“Social Interaction,” “Planning,” and other important human factors.

5. “AI technologies evolution as Cognitive Computing.” The AI technology evolution is on its way and growing in exponential form as a part of this evolution, based on the availability of new algorithms that need more specialized resources, as indicated in Chapter 1, Fig. 1.3. Each AI technology can be seen as a set of algorithms that give a special way of behaving to a smart system; these are: “Artificial intelligence (AI),” “Machine Learning (ML),” “Deep Learning (DL),” “Cognitive Computing (CC),” and others that are on their way.

Note*: “Cognitive Computing” as a subfield of “Cognitive Science” has the main objective to study, learn, and “simulate human thought process by AI computerized models.” Using “self-learning algorithms of ML and DL” one can apply data mining, pattern recognition, natural language processing, and other techniques that will allow computer system to mimic how the human brain works, as explained in Chapter 2, Introduction to Cognitive Science, Cognitive Computing, and Human Cognitive Relation to Help in the Solution of AI Biomedical Engineering Problems, Section 2.4 and shown in Chapter 2, Fig. 2.5.

6. “AI-ML-DL-CC Models Obtainment including Cognitive models” allows the development of many applications from automation to advance analytics in many fields and “Biomedical Engineering.”

Note*: CC models are obtained from a combination of AI technologies that mimic the human thought processes, and they are the processes studied in Cognitive Science when trying to understand the human brain and how it functions.

7. “AI-ML-DL-CC Human Functionalities analysis including Cognitive Evaluations,” These analysis techniques allow one to assist, verify, compare, and automate the human functionalities as a way to improve the synergy of human functionalities and AI.

Note*: “Cognitive Evaluations” are based on “Cognitive Assessments” [7], which evaluates areas of the brain function, including memory, concentration, processing speed, language, and reasoning capabilities. “Baseline cognitive assessment” provides a reference point to measure against in the case of problems or concerns (including major illness or brain injury) [8].

8. “Inference Engine” is a tool used to make logical deductions and derive conclusions about knowledge assets. They are useful in working with all sorts of information, for example, to enhance “Biomedical Engineering applying Artificial Intelligence.”
9. “Knowledge Storage AI Database (KSAID)” is defined as an organized collection of structured or

unstructured data in a database, with the purpose of locating fast data history, and to resolve further situations based on previous experiences. All the information stored is available to data management technologies in the system using smart relational databases, which allow interactive responses to the analysis goal prompted for users, which can be from humans, computer personal assistants, chatbots, robots, etc. They can be used in limitless “AI models” in applications such as “Natural Language Processing (NLP)” for information retrieval, information extraction, language translation, language spelling and grammatical accuracy of texts in text processors, text simplification, interactive voice response or IVR, sentiment analysis, text summarization, computer personal assistant, spam filters, speech recognition, natural language generation, speech generation, and many others as explained in more detail in Chapter 2, Section 2.5. In summary, “KSAID” is a technology to store complex structured and unstructured information on large relational databases that uses “Artificial Intelligence” tools to continuously organize and extract knowledge when needed that can be used by an “Inference engine” along with information that can be used for different types of reasoning based on the facts deduced or to find and highlight inconsistencies [9].

10. “Actions Generation” are the data results that can generate orders or procedures to be executed as actions in output devices, and usually are stored in the “knowledge storage database” as a new data by the “inference engine,” where the result is based on “affirmative reasoning results” that extract new data to reach a “positive inference.” This contrasts with the “denied reasoning results” based on inconsistencies detected as “negative inference.”
11. “Speech Generation” is an indispensable tool for output used for “Natural Language Understanding (NLP)” using “Cognitive Computing” and communicating the resulting “Cognitive AI models.”
12. “Image Generation” is an “AI tool” frequently used in images and videos to apply analysis of different variations, to add objects as augmented reality, etc.
13. “Motion Generation” is frequently used in many AI devices: for rehabilitation in healthcare, robots, and biomedical instruments, such as in MRI to move the magnetic field around and adjust the table the patient bed in order to obtain the axial/coronal/sagittal slices needed during the scanning, etc.

In this book the proposed “Cognitive Computing Agents System (AI-CCAS) framework” is used to analyze “nonmotor
(Continued)

(Continued)

symptoms of neurologic diseases” applying “Cognitive Learning Reasoning (CL&R) using Cognitive Computing (CC).” Also, this framework can be used to analyze “motor-symptoms of neurological diseases” and their research projects, as explained in my book: “Applied Biomechanics using mathematical models—Chapter 7 Case studies of applied Biomechanics solutions based on mathematical models” [10] with research of body movement disorders for different neurologic diseases based on the type of gait.

7.3 Inference engine and research example

“Inference Engine” is a tool used to make logical deductions and derive conclusions about knowledge assets. It applies logical rules to the “Knowledge Storage AI Database” to deduce new information typically represented as rules, like the “inference engine typical rule format” as represented in Eq. (7.1).

Inference Engine typical rule format

$$\begin{aligned} \text{IF } <\text{logical expression query}> \text{ THEN} \\ & <\text{logical expression deducted}> \end{aligned} \quad (7.1)$$

This is an iterative process of many logical rules, where each iteration can trigger additional rules to be queried in the information to reach a valid deduction or reject it. “Inference Engine” basically works in two modes: “forward reasoning” and “backward reasoning,” where:

- “Forward reasoning” is a logical inference of deductive arguments and rules of inferences.
- “Backward reasoning” is an inference method that applies a “depth-first-search algorithm” that starts in a tree in one selected arbitrary node known as the “root” and explores as far as possible along each branch before “backtracking.”

There are many ways to define “inference engine rules,” because they have to extract information based on imprecisions and uncertainties, such as symptoms that may or may not occur in patients, to reach a categorization needed for supervised models that can be applied to obtain “AI models,” as explain in the next Research 7.1.

7.3.1 Research 7.1

“Inference engine” to extract text information stored in the “Knowledge Storage AI Database” for “COVID-19 (SARS-COV-2)” symptoms and to define their categories. Note: This is an introductory example for understand an “inference engine” and its relation with “Knowledge Storage AI Database.”

Case for research

Define steps for an “Inference Engine” from a “General Architecture framework of an AI and Cognitive Computing Agents System (AI-CCAS)” to extract text information stored in the “Knowledge Storage AI Database” for “COVID-19 (SARS-COV-2)” symptoms, and to define the “categories (classes)” to be used to obtain an “AI model.”

General Objective

Obtain and apply the general steps needed for an “Inference Engine” to extract text information from the “Knowledge Storage AI Database” in the “AI and Cognitive Computing Agents System (AICCAS).”

Specific Objectives

- Define the general step for an “Inference Engine” to extract text information from the “Knowledge Storage AI Database” for “COVID-19 (SARS-COV-2)” symptoms.
- Introduce “fuzzy rule extraction methods” for the “Inference Engine” of the “General Architecture framework of AI and Cognitive Computing Agents System (AI-CCAS).”
- Develop the “inference engine criteria” for the “coronavirus COVID-19 symptoms.”
- Identify the variables needed for the “fuzzy inputs X ” and “fuzzy outputs Y ” to define their “fuzzy universal set” and “fuzzy sets.”
- Obtain the “Phenomenon fuzzy sets criteria” that represents the summary of the variables needed to describe the symptoms and their category classes.
- Define the “fuzzy input linguist variable” and their corresponding “fuzzy output linguist variable” to define the “fuzzy rules” needed to extract the text information.
- Extract text information as a useful dataset using the “inference engine” to obtain the results in a specific dataset from the “Knowledge Storage AI Database.”

Background for “COVID-19 symptoms”

“COVID-19” affects different people in different ways. Most infected people will develop mild to moderate illness and recover without hospitalization. Anyone can have mild to severe symptoms. But older adults and people who have severe underlying medical conditions like heart or lung disease or diabetes seem to be at higher risk for developing more serious complications from “COVID-19.” The symptoms are sometimes confusing based in the fact that people with COVID-19 have had a wide range of symptoms reported: ranging from mild symptoms to severe illness, and symptoms may appear 2–14 days after exposure to the virus. People with these symptoms may have COVID-19 [11,12].

Dataset

The dataset for coronavirus symptoms can be extracted from different well recognized “health web-sites,” such as <http://www.cdc.gov> [11], <http://www.who.int> [12], etc., as shown in Fig. 7.2.

<https://www.cdc.gov/coronavirus>

Watch for symptoms

People with COVID-19 have had a wide range of symptoms reported – ranging from mild symptoms to severe illness. Symptoms may appear 2-14 days after exposure to the virus. People with these symptoms may have COVID-19:

- Fever or chills
- Cough
- Shortness of breath or difficulty breathing
- Fatigue
- Muscle or body aches
- Headache
- New loss of taste or smell
- Sore throat
- Congestion or runny nose
- Nausea or vomiting
- Diarrhea

This list does not include all possible symptoms. CDC will continue to update this list as we learn more about COVID-19.

When to seek emergency medical attention

Look for **emergency warning signs*** for COVID-19. If someone is showing any of these signs, **seek emergency medical care immediately**:

- Trouble breathing
- Persistent pain or pressure in the chest
- New confusion
- Inability to wake or stay awake
- Bluish lips or face

*This list is not all possible symptoms. Please call your medical provider for any other symptoms that are severe or concerning to you.

Call 911 or call ahead to your local emergency facility. Notify the operator that you are seeking care for someone who has or may have COVID-19.

<https://www.who.int/health-topics/coronavirus>

COVID-19 affects different people in different ways. Most infected people will develop mild to moderate illness and recover without hospitalization.

Most common symptoms:

- fever.
- dry cough.
- tiredness.

Less common symptoms:

- aches and pains.
- sore throat.
- diarrhoea.
- conjunctivitis.
- headache.
- loss of taste or smell.
- a rash on skin, or discolouration of fingers or toes.

Serious symptoms:

- difficulty breathing or shortness of breath.
- chest pain or pressure.
- loss of speech or movement.

Seek immediate medical attention if you have serious symptoms. Always call before visiting your doctor or health facility.

People with mild symptoms who are otherwise healthy should manage their symptoms at home.

On average it takes 5–6 days from when someone is infected with the virus for symptoms to show, however it can take up to 14 days.

FIGURE 7.2 Dataset from different “health websites” as: <http://www.cdc.gov> and <http://www.who.int>.

Procedure

Implement the “*inference engine*” to extract text information stored or to be stored in the “*Knowledge Storage AI Database*” for “*COVID-19 (SARS-COV-2)*” symptoms, and define their categories for the “*AI model*.” The “*fuzzy inference systems*” defined in “*Soft Computing*” allows one to obtain computing models that are tolerant of the imprecision and uncertainty and that work with a partial truth under approximation to achieve tractability and robustness with a low computing solution cost. Because usually the text information is vague and imprecise one common approach is to apply one of the “*fuzzy rule extraction*” methods. The general steps for an “*inference engine*” under “*fuzzy inference systems*” are:

1. Obtain from websites or the “*Knowledge Storage AI Database*” the current medical procedures for the disease; they are explained in “*disease risk assessments*,” with the purpose of developing the “*inference engine criteria*” for the disease “*COVID-19 (SARS-COV-2 virus)*,” that is, SARS-CoV-2 Sample Type Risk Assessment [13], Multistate Assessment of SARS-CoV-2 Seroprevalence in Blood Donors [14], Overview of Testing for SARS-CoV-2 (COVID-19) [15], etc.
2. Identify the variables needed for the “*fuzzy inputs X*” and “*fuzzy outputs Y*” to define their “*fuzzy universal set*” and “*fuzzy sets*.” Define their “*fuzzy linguist variables: inputs (I_x), outputs (o_y)*” and their respective “*input linguist variables (IL_j)*” and “*output fuzzy*

membership functions (OL_j).” To achieve this, build a table of “*phenomenon fuzzy sets criteria*” that represent the summary of the variables needed, such as medical status (I_x), text definition and or values (IL_j), and outputs (o_y) and their respective linguist output or category, as shown in Table 7.1

3. Take the “*input linguist variable*” and their corresponding “*output linguist variable*” to define the “*fuzzy rules*” to be applied based on the fuzzy variables shown in Eq. (7.2).

“*General fuzzy rule inference engine for information extraction*”

$$IF I_x \text{ is } IL_j \text{ THEN } O_y \text{ is } OL_y \quad (7.2)$$

Applying Eq. (7.2) to the data of Table 7.1, we can define the necessary “*fuzzy rules*” to extract the information based on each medical marker to obtain their related category, as shown in Table 7.2 for the “*Inference Engine*” in the “*AI-CCAS (AI & Cognitive Computing Agents System)*.”

Note*: This dataset will be used in the next research example 7.2

4. Extract text information using the iterative “*Fuzzy rules*” of the “*inference engine*” applying steps 1–3 to obtain the results to be stored in a specific dataset* in the “*Knowledge Storage AI Database*.”
5. Apply an “*AI algorithm**” as shown in the next “*research 7.2 Attention Network*” as an example of a “*Long Short-Term Memory*” Neural Network to Classify Text of COVID-19 (SARS-COV-2) symptoms.

TABLE 7.1 “Phenomenon fuzzy sets criteria” that represent a summary of the variables needed for COVID-10 (SAR-CoV-2 virus).

Medical_status ($I_x = \text{input variable}$)	Markers values ($IL_j = \text{input linguist variable}$)	Category ($OL_y = \text{linguist output}$)
Recommendation	Social distance, Stay 6 feet away, keep distance, stay away from respiratory products, asymptomatic, wash hands often, avoid direct contact, mask, cover mouth, cover nose, monitor health, cover sneeze, check temperature, gloves, clean, disinfect, gel antibacterial, stay home, stay in touch with doctor, avoid public crowded public places, sleep well, isolate when symptoms, do not share drinks, medicines on time, medicines availability, check blood pressure, barrier shield, remote shopping, eat healthy food, drink water, drink electrolyte, separate from ill people, soap, hand sanitizer, first aid essentials, paper towels, write down symptoms, emergency contact, avoid pets interaction	Prevention
Low–medium risk	Fever, chill, cough, fatigue, cough, muscle aches, body aches, headache, loss of taste, loss of smell, sore throat, nasal congestion, runny nose, nausea, vomiting, diarrhea	Symptoms
Medium–high risk	65-year-old or older, living nursing home, underlying medical conditions, chronic lung disease, moderate asthma, severe obesity, serious heart conditions, diabetes, chronic kidney disease	High risk
High risk	Trouble breathing, pain, pressure in chest, confusion, inability stay awake, call 911, call ahead emergency facilities, inflammatory syndrome, damage the respiratory system, liver disease, stressed, immunocompromised conditions, asthma, hemoglobin disorders, difficulty sleeping or concentrating	Emergency care
Urgent	Inability to wake up, bluish lips, bluish face, heart failure, shortness of breath, difficulty breathing, Persistent pain or pressure in the chest, New confusion,	Urgent higher risk

TABLE 7.2 Fuzzy rules* to extract text information stored or to be stored in the “Knowledge Storage AI Database” for “COVID-19 (SARS-COV-2).”

Rules	Inference engine fuzzy rules for information extraction
1	IF “ $I_{x=1}$ ” = “recommendation” is “ $IL_{j=1}$ ” = “social distance” THEN “ $OL_{y=1}$ ” = “Prevention”
...	... (Repeat this rule “ $I_{x=1}$ ” = all IL_j for “ $OL_{y=1}$ ”)
2	IF “ $I_{x=2}$ ” = “Low–medium risk” is “ $IL_{j=1}$ ” = “fever” THEN “ $OL_{y=1.25pt} = -1.25pt2$ ” = “Symptom”
...	... (Repeat this rule “ $I_{x=2}$ ” = all IL_j for “ $OL_{y=2}$ ”)
3	IF “ $I_{x=3}$ ” = “Medium–high risk” is “ $IL_{j=1}$ ” = “ ≥ 65 -year-old” THEN “ $OL_{y=3}$ ” = “High risk”
...	... (Repeat this rule “ $I_{x=3}$ ” = all IL_j for “ $OL_{y=3}$ ”)
4	IF “ $I_{x=4}$ ” = “High risk” is “ $IL_{j=1}$ ” = “trouble breathing” THEN “ $OL_{y=4}$ ” = “Emergency care”
...	... (Repeat this rule “ $I_{x=4}$ ” = all IL_j for “ $OL_{y=4}$ ”)
5	IF “ $I_{x=5}$ ” = “Urgent” is “ $IL_{j=1}$ ” = “inability to wake up” THEN “ $OL_{y=5}$ ” = “Urgent Higher risk”
...	... (Repeat this rule “ $I_{x=5}$ ” = all IL_j for “ $OL_{y=5}$ ”)

Note*: Fuzzy rules extraction will be explained with more details and examples in Section 7.6.7 “Neuro-Fuzzy Logic Reasoning.”

Conclusions

In this research the theory for an “inference engine” shows that it is possible to extract text information stored in the “Knowledge Storage AI Database” for “COVID-19 (SARS-COV2)” symptoms and to define their categories. The steps to achieve this goal are implemented in the next research 7.2 Attention network for NLP applying Long Short-Term Memory to Classify Text of COVID-19 (SARS-COV2) symptoms.

7.3.2 Research 7.2

“Attention network” for “NLP applying long short-term memory” to classify text of COVID-19 (SARS-COV2) symptoms.

Case for research

“Obtain an Attention network for NLP applying an LSTM model from MATLAB® Deep Learning Toolbox using a Deep Network Designer to classify and then predict from text descriptions COVID-19 symptoms obtained from the “Knowledge Storage AI Database (KSAID)” as a component of the framework: Cognitive Computing Agents System (AI-CCAS).

General Objective

Apply “MATLAB Deep Learning Toolbox” to define, build, train, and deploy an “Attention network for NLP” applying an “LSTM Neural Network (LSTM NN) model” to classify text descriptions for “COVID-19

(SARS-COV2),” and to predict new descriptions of symptoms in patients.

Specific Objectives

- Load the “Covid19Repts.csv” dataset that contains two fields: descriptions and categories.
- Create two partitions for: training with 70% and validation 30% of the text dataset and visualize the “Word-cloud bag” form each partition.
- Preprocess the text, tokenizing both text partitions, as explained in Chapter 2
 - Convert the text from both partitions to “Sequences” and “Indices” to be used in the “LSMT NN” to process the text accordingly.
- Define, create, visualize, analyze, and train the custom “LSMT NN” using the MATLAB “Deep Network Designer” available in the MATLAB Deep Learning Toolbox.
- Classify new text descriptions for COVID-19 symptoms to predict their categories or classes.

Background for “COVID-19 symptoms”

Read Background for “COVID-19 symptoms” from Research 7.1.

Dataset

The “Covid19Repts.csv” dataset is a dataset recompilation of text descriptions of COVID-19 extracted from “Knowledge Storage AI Database” using the “Inference engine” explained in research 7.1, organized into five categories: “preventions,” “symptoms,” “high risk,” “emergency care,” and “urgent higher risk,” as shown in Fig. 7.3.

Description	Category
Fever or chills	Symptom
Cough	Symptom
Shortness of breath or difficulty breathing	urgent higher risk
Fatigue	Symptom
Muscle or body aches	Symptom
Headache	Symptom
New loss of taste or smell	Symptom
Sore throat	Symptom
Congestion or runny nose	Symptom
Nausea or vomiting	Symptom
Diarrhea	Symptom
Trouble breathing	emergency care
Persistent pain or pressure in the chest	emergency care
New confusion	emergency care
Inability to wake or stay awake	emergency care
Bluish lips or face	emergency care
Stay 6 feet away	Prevention
stay away of respiratory droplets produced when an infected person coughs, sneezes or talks.	Prevention
avoid droplets that can land in the mouths or noses of people who are nearby or possibly be inhaled into the lungs.	Prevention
may be spread by people who are not showing symptoms.	Prevention
Wash your hands often	Prevention
Avoid close contact	Prevention
Cover your mouth and nose with a cloth face cover when around others	Prevention
Cover coughs and sneezes	Prevention
Monitor Your Health	Prevention
Take your temperature if symptoms develop	Prevention
Use of Cloth Face Coverings to Help Slow the Spread	Prevention
Use wear gloves	Prevention
Cleaning And Disinfecting Your Home	Prevention
Keep Your Distance to Slow the Spread	Prevention
People 65 years and older with Symptoms SAR COV 2	High risk
People who live in a nursing home or long-term care facility with Symptoms SAR COV 2	High risk
People of all ages with underlying medical conditions, particularly if not well controlled with Symptoms SAR COV 2	High risk
People with chronic lung disease or moderate to severe asthma	High risk

FIGURE 7.3 Dataset sample “Covid19Repts.csv.”

TABLE 7.3 Dataset “Covid19Reports.csv” fields and descriptions.

Field	Description * instances of text descriptions for COVID-19 symptoms. Num. of fields = 2
Description	String of text up to 250 characters
Category	Classes = [“preventions,” “symptoms,” “high risk,” “emergency care,” and “urgent higher risk”]

Note*: This data set is available in the companion directory of the book, in the following directory: “. . . \Exercises_book_ABME\CH7\MATLAB_LSTM_Text\Covid19Reports.csv.”

The “Covid19Reports.csv” dataset has a structure with two columns, as shown in [Table 7.3](#).

Procedure

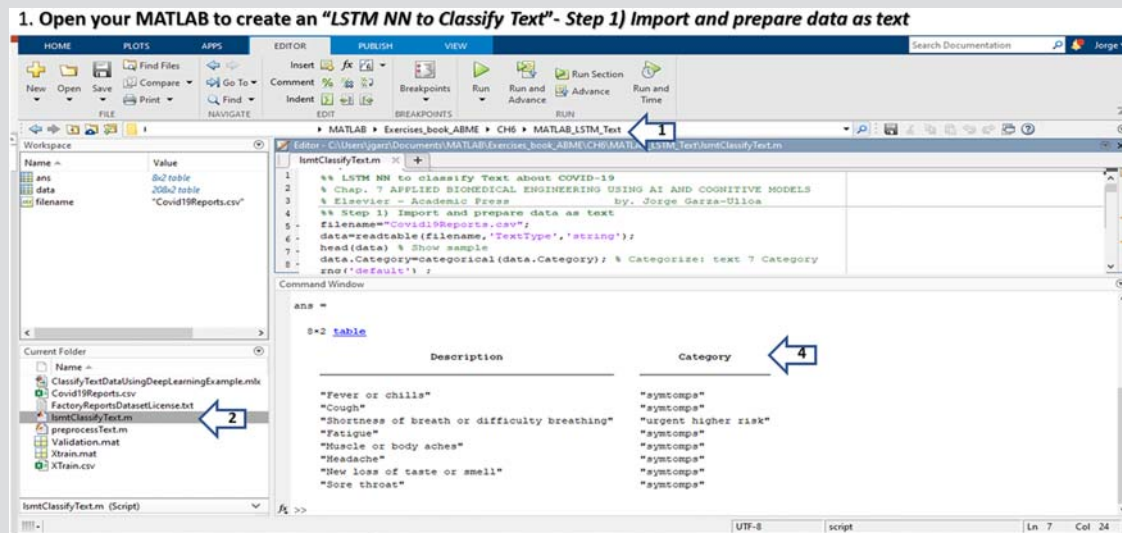
The steps to “Obtain an AI model Attention network using an LSTM model from MATLAB Deep Learning

Toolbox using a Deep Network Designer to classify and then predict from text descriptions for COVID-19 symptoms” are summarized in Table of slides 7.1 + 7.2, and each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

Table of slides 7.1 + 7.2 Steps for MATLAB Deep Learning Toolbox to obtain an Attention network using an LSTM model to classify text.

Slide Description Screen figure

1 Open your MATLAB to “create an AN-LSTM NN to Classify Text”—Step (1) Import and prepare data as text. Go to the directory “... \Exercises_book_ABME\CH7 \MATLAB_LSTM_Text.” Open the script “IsmtClassifyText.m.” Copy and paste in the command prompt: “step (1) Import and prepare data as text.” And show in the screen a sample from the categorization of the data text.



Go to the directory “... \Exercises_book_ABME\CH7 \MATLAB_LSTM_Text”. Open the script “IsmtClassifyText.m”. Copy and paste in the command prompt: “step 1) Import and prepare data as text”. And show in the screen a sample from the categorization of the data text

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 2 Description
MATLAB “create an LSTM NN to Classify Text”—Step (2) Partition data for Training —Validation and visualize them.
Copy and paste in the command prompt “step (2) Partition data for Training —Validation and visualize partitions.” And obtain the “Word-cloud bag” figures for “Training Data” and “Validation Data.” Note: Dismiss the warnings for class value in partitions.

Screen figure

2. MATLAB “create an LSTM NN to Classify Text”- Step 2) Partition data for Training - Validation and visualize them

The screenshot displays the MATLAB IDE with the following components:

- Workspace:** Lists variables such as 'ans', 'cvp', 'data', 'dataTrain', 'dataValidation', 'filename', 'textDataTrain', 'textDataValidation', 'Validation', and 'YTrain' with their respective data types.
- Editor:** Contains MATLAB code for Step 2, including data partitioning and visualization commands. A blue arrow labeled '1' points to the `figure; wordcloud(textDataTrain); title("Training Data");` line.
- Command Window:** Shows a warning message: "Warning: One or more either remove this cell with corresponding reproc" and "In internal_data.c In internal_data.c In internal_data.c In internal_data.c". A blue arrow labeled '2' points to the first instance of this warning.
- Visualizations:** Two word cloud plots are shown. The left plot is titled "Training Data" and features prominent words like "People", "Symptoms", "COV", "disease", "care", "get", "medical", "local", "cough", "breathing", "icon", "care", "get", "medical", "local", "cough", "health", "risk", "COVID-19", "Avoid", "members", "blood", "Business", "emergency", "office", "household". The right plot is titled "Validation Data" and features prominent words like "Severe text", "Wash face", "Clean", "Cloth", "hands", "COV", "stay", "SAR", "icon", "Cover", "Take", "feel", "staff", "chronic", "coughs", "People", "inability", "surfaces", "Symptoms", "sick", "paper", "distress", "soap", "disinfect", "wake", "confusion", "called", "immune", "lowells", "healthy", "nose", "serious".

Copy and paste in the command prompt “step 2) Partition data for Training - Validation and visualize partitions”. And obtain the “Word-cloud bag” figures for “Training Data” & “Validation Data”. Note: Dismiss the warnings for class value in partitions.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 3 MATLAB “create an LSTM NN to Classify Text”—Step (3) Preprocess text from training and validation. Copy and paste in the command prompt “step (3) Preprocess text from training and validation” and see samples of “tokenized text for training and validation.”

3. MATLAB “create an LSTM NN to Classify Text”- Step 3) Preprocess text from training and validation

The screenshot displays the MATLAB environment with the following components:

- Workspace:** Lists variables such as `ans` (5x1 tokenized document), `cvp` (1x1 cvpartition), `data` (208x2 table), `dataTrain` (146x2 table), `dataValidation` (62x2 table), `documentsTrain` (146x1 tokenized document), `documentsValidation` (62x1 tokenized document), `filename` (146x1 string), `textDataTrain` (146x1 string), `textDataValidation` (62x1 string), `Validation` (62x1 categorical), and `YTrain` (146x1 categorical).
- Editor:** Contains MATLAB code for Step 3:


```

%% Step 3) Preprocess text from training and validation.
documentsTrain = preprocessText(textDataTrain);
documentsValidation = preprocessText(textDataValidation);
% View the first few preprocessed
disp("Sample of text preprocessed for training"); documentsTrain(1:5)
disp("Sample of text preprocessed for validation"); documentsValidation(1:5)
% See Step 4) Convert Document to Sentences and Indices needed as Input of LSTM network
      
```
- Command Window:** Shows the execution output:


```

s=1 tokenizedDocument:
3 tokens: fever or chills
1 tokens: cough
6 tokens: shortness of breath or difficulty breathing
1 tokens: fatigue
4 tokens: muscle or body aches

Sample of text preprocessed for validation

ans =
s=1 tokenizedDocument:
1 tokens: headache
2 tokens: sore throat
4 tokens: congestion or runny nose
1 tokens: diarrhea
6 tokens: inability to wake or stay awake
      
```

Copy and paste in the command prompt “step 3) Preprocess text from training and validation” and see samples of “tokenized text for training and validation”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

4 MATLAB “create an LSTM NN to Classify Text”—Step 4) Convert Document to Sequences and indices as input of LSTM network. Copy and paste in the command prompt “step 4) Convert Document to Sequences and indices as input of LSTM network,” obtain a figure for the “training document lengths,” showing that the most are 10 or lower. And list a sample from indices from the sequences.

4 . MATLAB “create an LSTM NN to Classify Text”- Step 4) Convert Document to Sequences and indices as input of LSTM network

Copy and paste in the command prompt “step 4) Convert Document to Sequences and indices as input of LSTM network”, obtain a figure for the “training document lengths”, showing that the most are 10 or lower. And list a sample from indices from the sequences.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

5

MATLAB “create an LSTM NN to Classify Text”—Step (5) Define, create, visualize, analyze, train and create a LSTM model.

Copy and paste in the command prompt “step (5) Define, create, visualize, analyze, train and create a LSTM model,” define the layer to be used in our net, call the “deepNetworkDesigner” as shown in the next slide.

5. MATLAB “create an LSTM NN to Classify Text”- Step 5) Define, create, visualize, analyze, train and create a LSTM model

```
function net = trainLstmClassifyText(XTrain, YTrain, layers, options)
% Step 5) Define, create, visualize, analyze, train and create a LSTM model
inputSize = 1; embeddingDimension = 50; numHiddenUnits = 50;
numWords = endNumWords; numClasses = numel(categories(YTrain));
layers = [ ...
    sequenceInputLayer(inputSize)
    wordEmbeddingLayer(embeddingDimension, numWords)
    lstmLayer(numHiddenUnits, 'OutputMode', 'Last')
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer];
% Visualize and analyze layer with deepNetworkDesigner
deepNetworkDesigner
% From in the Deep Designer from workspace select layers and analyze them
options = trainingOptions('adam', ...
    'MiniBatchSize', 16, ...
    'GradientThreshold', 5, ...
    'Shuffle', 'every-epoch', ...
    'ValidationData', {XValidation, YValidation}, ...
    'Plot', 'training-progress', ...
    'Verbose', false);
net = trainNetwork(XTrain, YTrain, layers, options);
end
```

```
>> net = trainNetwork(XTrain, YTrain, layers, options);
>>
```

```
class layers array with layers:
 1 ** Sequence Input      Sequence input with 1 dimensions
 2 ** Word Embedding Layer Word embedding layer with 50 dimensions and 518 unique words
 3 ** LSTM                LSTM with 50 hidden units
 4 ** Fully Connected     5 fully connected layer
 5 ** Softmax             Softmax
 6 ** Classification Output CrossEntropy
```

Copy and paste in the command prompt “step 5) Define, create, visualize, analyze, train and create a LSTM model”, define the layer to be used in our net, call the “deepNetworkDesigner” as shown in the next slide.

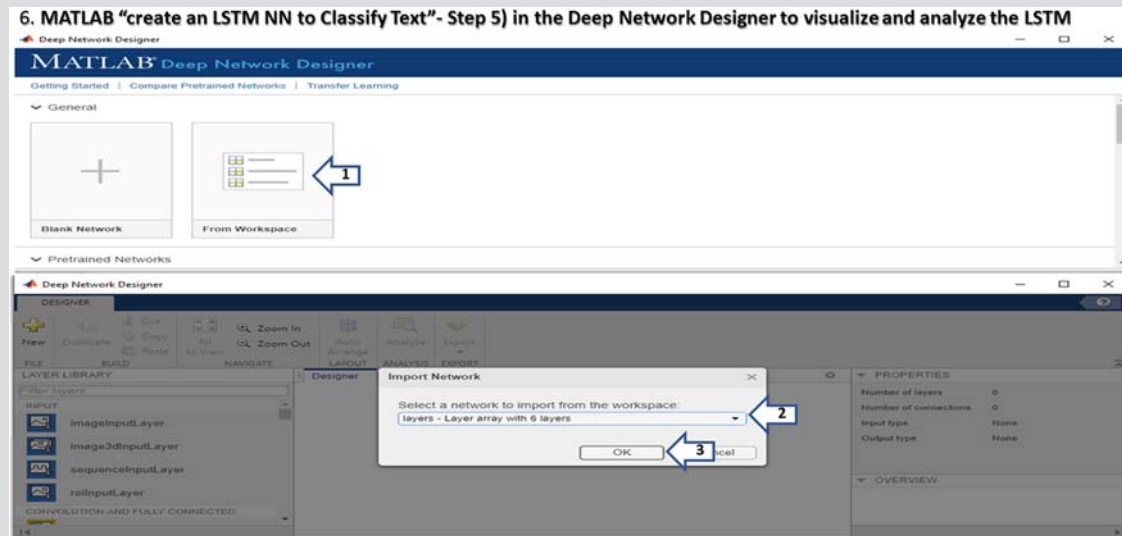
From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide 6 Description
MATLAB “create an LSTM NN to Classify Text”—Step (5) in the Deep Network Designer to visualize and analyze the LSTM.
Select “.From Workspace” in the first screen of “Deep Network Designer,” then select “Layer array with 6 layers” to be imported from the workspace as shown in the lower picture, and click the “OK” button.

Screen figure



Select “From Workspace” in the first screen of “Deep Network Designer”, then select “Layer array with 6 layers” to be imported from the workspace as shown in the lower picture, and click the “OK” button

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

7

MATLAB “create an LSTM NN to Classify Text”—Step (5) in the Deep Network Designer to visualize and analyze the LSTM.

In the “Deep Network Designer” explore the “net” loaded from the workspace, click on “lstmLayer” and verify that “NumHiddenUnits = 80.” Then click on the “Analyze” Icon as indicated in the slide.

7. MATLAB “create an LSTM NN to Classify Text”- Step 5) in the Deep Network Designer to visualize and analyze the LSTM

The screenshot displays the MATLAB Deep Network Designer interface. The top toolbar contains an 'Analyze' button (indicated by arrow 4). The main workspace shows a neural network architecture with the following layers: sequenceinput, word-embedd..., lstm (lstmLayer, indicated by arrow 2), fc fullyConnected..., softmax softmaxLayer, and classoutput classificationLa... The Properties panel on the right shows the configuration for the selected 'lstmLayer', with 'NumHiddenUnits' set to 80 (indicated by arrow 3). The Layer Library on the left lists various layer types under categories like 'INPUT', 'CONVOLUTION AND FULLY CONNECTED', and 'SEQUENCE'.

In the “Deep Network Designer” explore the “net” loaded from the workspace, click on “lstmLayer” and verify that “NumHiddenUnits=80”. Then click on the “Analyze” Icon as indicated in the slide.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

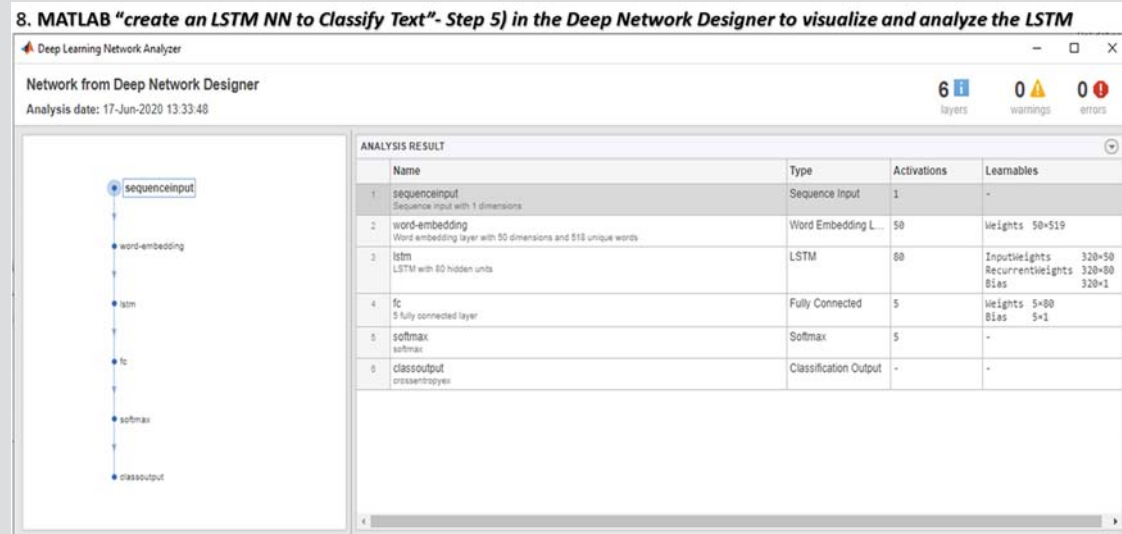
(Continued)

(Continued)

Slide Description

Screen figure

8 MATLAB “create an LSTM NN to Classify Text”—Step (5) in the Deep Network Designer to visualize and analyze the LSTM. The “Deep Learning Network Analyzer” must show the “6 layers with 0 earnings and 0 errors.” Then select the screen for the “Training progress.”



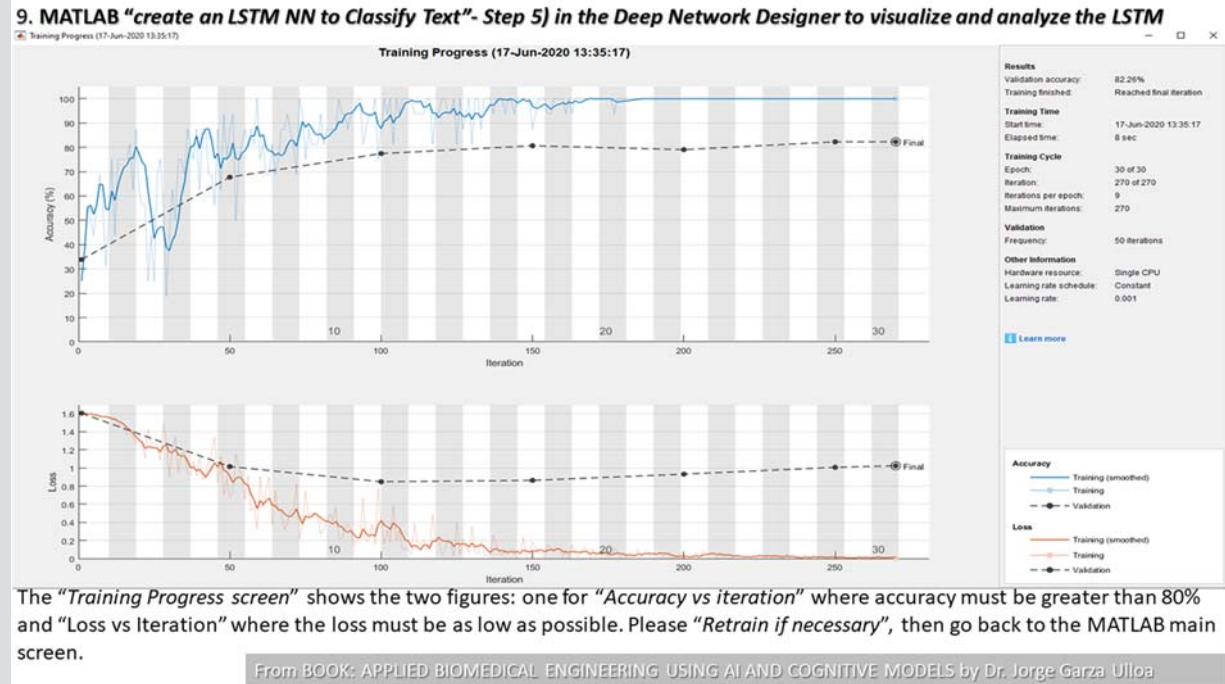
The “Deep Learning Network Analyzer” must show the “6 layers with 0 earnings and 0 errors”. Then select the screen for the “Training progress”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

9

MATLAB “create an LSTM NN to Classify Text”—Step (5) in the Deep Network Designer to visualize and analyze the LSTM.

The “Training Progress screen” shows the two figures: one for “Accuracy versus iteration” where accuracy must be greater than 80% and “Loss versus iteration” where the loss must be as low as possible. “Retrain if necessary,” then go back to the MATLAB main screen.



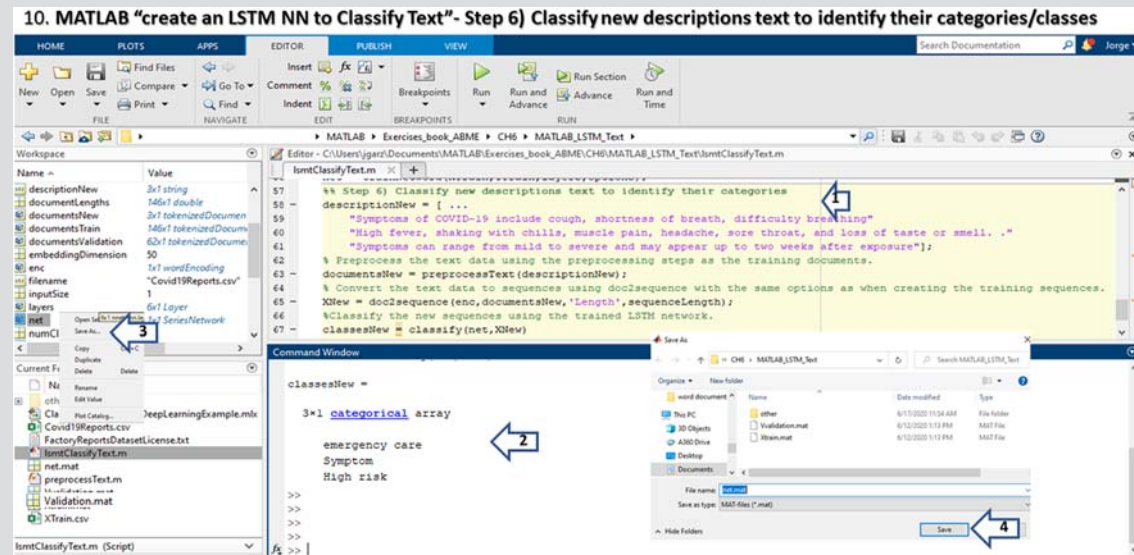
(Continued)

(Continued)

Slide Description

Screen figure

10 MATLAB “create an LSTM NN to Classify Text”—Step (6) Classify new descriptions text to identify their categories/classes.
Copy and paste in the command prompt “step (6) Classify new descriptions text to identify their categories/classes” to obtain the classes for the 3 new text analyzed as indicated. Finally save to the workspace the “net model” with right click to be used for others text predictions. Close all and exit MATLAB.



Copy and paste in the command prompt “step 6) Classify new descriptions text to identify their categories/classes” to obtain the classes for the 3 new text analyzed as indicated. Finally save to the workspace the “net model” with right click to be used for others text predictions. Close all and exit MATLAB.

From BOOK APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

“An LSTM model from MATLAB Deep Learning Toolbox using a Deep Network Designer to classify and then predict from text descriptions for COVID-19 symptoms” was obtained with excellent results for the deduction of their categories or classes.

Recommendation

This kind of “LSTM model” can be applied to apply to a diversity of “NLP” problems to analyze specific topics in Biomedical Engineering.

7.4 Action generation

“Actions Generation” are the data results to generate orders or procedures to be executed as actions in output devices that are to be delivered to the patient through reports in different ways, such as “Speech Generation,” “Image Generation,” and “Motion Generation.”

infrastructure with the purposes of driving changes, eliminating inefficiencies, enabling quick adaptations to the market, supplying changes, and allowing growth in the correct direction. The same “BI” concept “Artificial Continue Intelligence in real time” is applicable to “Business Intelligence in Healthcare,” as the process by which large-scale data from the healthcare industry is now collected and refined into actionable insights mainly in many different healthcare areas, such as to cut costs, select medicines, follow-up patient care, clinical data analysis, patient behavior prediction, faster research, better education, etc., to obtain invaluable performance in healthcare, that allows acting on real-time data for real-life situations saving lives. “Business Intelligence in Healthcare” can help basically in five domains: “Patient Engagement,” “Care Delivery,” “Population Health,” “R&D,” and “Administration” [16]

In the same way, we can apply in real time “Artificial Continue Intelligence” of “Business Intelligence in

“Actions Generation” are apparently the last step of the process as shown in Fig. 7.1 “Cognitive Computing Agents System (AI-CCAS),” but in reality it is only the beginning of

another loop for the “Artificial continue Intelligence” framework that typically works in real time, running a kind of loop as explained and indicated in Fig. 7.4.

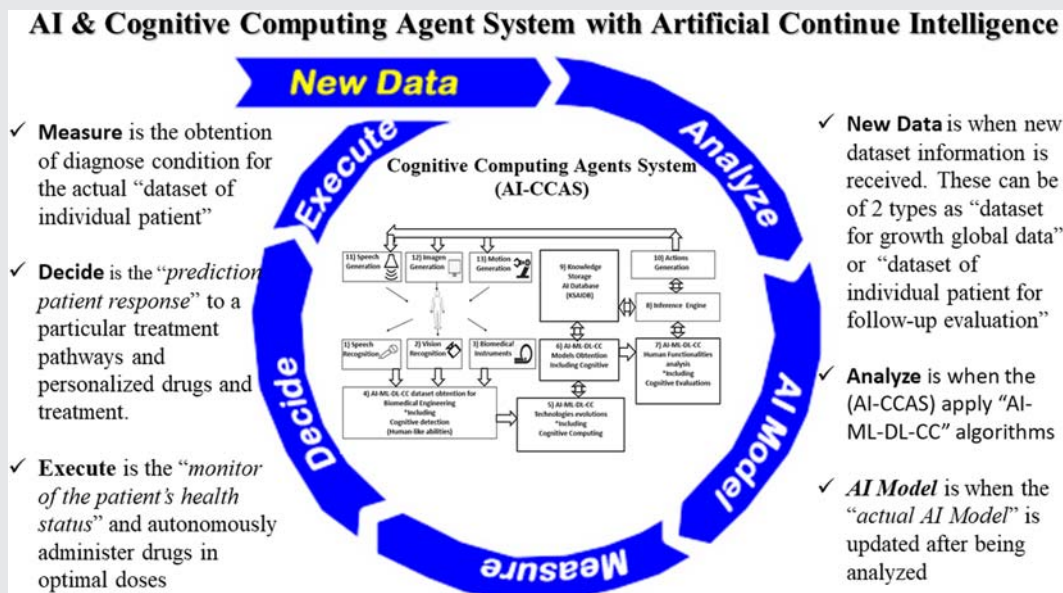


FIGURE 7.4 AI and Cognitive Computing Agent System with artificial continue intelligence in the continuous process at the “AI-CCAS”.

7.5 Business intelligence in healthcare

“Artificial Continue Intelligence” is a schema to use analytics of “AI Models in real time” to be integrated in different processes generally known as “business intelligence (BI).” Where “BI” combines business analytics, data mining, data visualization, data tools, and

Healthcare” in the “AI and Cognitive Computing Agents System (AI-CCAS),” as explain in Section 7.2 and shown in Fig. 7.4, to continuously evaluate different “human illnesses,” “human diseases,” and “many disorders.”

“AI and Cognitive Computing Agent System with Artificial Continue Intelligence” can process the patient’s information by applying the following steps when “New

Data” arrives: “*Analyze*,” “*AI Model*,” “*Measure*,” “*Decide*,” and “*Execute*,” where:

- “*New Data*” is when new dataset information is received, this can be in different formats, such as text data, image scanning, videos, etc. This can be received as a dataset from a “*dataset for growth global data*” to be incorporated in the “*Knowledge Storage AI Database (KSAID)*” to update the current “*AI model*” or as a “*dataset of individual patient to be evaluated*” with the actual “*AI model*.”
- “*Analyze*” is when the (AI-CCAS) applies “*AI-ML-DL-CC*” algorithms incorporating the new “*growth global data*.”
- “*AI Model*” is when the “*actual AI Model*” is updated after being analyzed.
- “*Measure*” is the obtainment of the condition diagnosis for the actual “*dataset of individual patient*,” classified using the “*AI model*,” for example, X-rays/MRI image examination and evaluation.
- “*Decide*” is the “*prediction patient response*” to particular treatment pathways, personalized drugs, and treatments.
- “*Execute*” is the “*monitoring of the patient’s health status*,” and autonomously administering the drug in optimal doses.

The proposed framework in this book “*AI and Cognitive Computing Agent System (AI-CCAS) with Artificial Continue Intelligence*” will allow the development of many applications that can help healthcare, such as:

- “*Patient data flagging*”: an indication of when patients are declining in strength.
- “*Patient intervention suggestions*”: suggests effective intervention strategies to improve patient’s health.
- “*Patient self-monitoring*”: automatic frequent reading to provide feedback of patient’s status.
- “*Condition management*”: helps individuals manage chronic conditions, which in turn drives down costs and improves compliance with their physician’s care plan.
- “*Patient flow optimization*”: it is a critical component of process management optimization in hospitals and other healthcare facilities.
- “*Clinical trial matching*”: it facilitates patient enrollment in clinical trials by identifying potential trials for interested patients and their caregivers and providers.
- “*Image Analysis*”: it is the extraction of meaningful information from digital images applying image AI processing techniques.
- “*Radiation plan design*”: in radiotherapy, “*radiation treatment planning (RTP)*” is the process in which a team consisting of radiation oncologists, radiation

therapists, medical physicists, and medical dosimetrists plan the appropriate external beam radiotherapy or internal brachytherapy treatment technique for a patient with cancer.

- “*Disease pattern identification*”: used in disease pattern identification.
- “*Drug regimen selection*”: a structured treatment plan designed to improve and maintain health.
- “*Medication administration*”: it is the applying, dispensing, or giving of drugs or medicines as prescribed by a physician.
- “*Care pathway identification*”: it is a complex intervention for the mutual decision-making and organization of care processes for a well-defined group of patients during a well-defined period.
- “*Adverse event prediction*”: it is the process of identifying potential adverse events of an investigational drug before they occur in a clinical trial.
- “*Asset demand prediction*”: the major demand forecasting assets demand methodologies based mainly on “*artificial neural networks (ANNs)*” and “*multiple regression methodologies*.”
- “*Risk prediction*”: risk prediction tools are developed to identify patients at risk and to facilitate physicians’ decision-making.
- “*Physical therapy*”: it is used to improve a patient’s physical functions through physical examination, diagnosis, prognosis, patient education, physical intervention, rehabilitation, disease prevention, and health promotion.
- “*Robotic-assisted surgery*”: allows doctors to perform many types of complex procedures with more precision, flexibility, and control than is possible with conventional techniques. Robotic surgery is usually associated with minimally invasive surgery procedures performed through tiny incisions.
- “*Cognitive evaluations*”: it is a theory in psychology that is designed to explain the effects of external consequences on internal motivation.
- “*Telehealth*” is health-related services and information via electronic information and telecommunication technologies, allowing long-distance patient and clinician contact, care, advice, reminders, education, intervention, monitoring, and remote admissions.
- And many more.

The “AI and Cognitive Computing Agent System with Artificial Continue Intelligence” can be used in many healthcare areas; in this chapter we focus on “Cognitive Learning and its relationship with the neuroscience of reasoning, proposed as Cognitive Learning- Reasoning (CL&R) using Cognitive Computing (CC).”

7.6 Learning and reasoning relationship of biomedical engineering, cognitive science, and computer science through artificial intelligent models

As explained in Chapter 1 of this book the main focus is the relationship between three different multidiscipline engineering branches: *Machine Learning, Deep Learning,* and “*Cognitive Learning and its relationship with neuroscience of reasoning, proposed as Cognitive Learning-Reasoning (CL&R) using Cognitive Computing (CC),*” in order to study how the nervous and musculoskeletal systems obey movements orders, and to understand how information is mentally processed in cognition when injuries and neurologic diseases are present in the human body. “*The aim is to find a solution that can help to understand, measure, and follow-up the diseases through AI models.*”

“*Cognitive science (CoSi)*” is the interdisciplinary scientific integration that studies the mind and its processes. It examines the nature, tasks, and the functions of human cognition, which is the process of acquiring knowledge and understanding through thought, experience, and the senses. *CoSi* is the objective of developing theories about human perception, action, memory, attention, reasoning, decision-making, language use, and learning. We will center on proposing projects to resolve “*Biomedical Engineering and Neurology*” problems by studying methodologies from “*Cognitive learning*” to obtain “*AI models*” applying the “*Computer Science*” tools of “*AI*” as “*Machine Learning,*” “*Deep Learning,*” and “*Cognitive computing,*” using techniques learned through this book’s chapters.

“*Cognitive learning*” is learning based on a sequence of processes: observing, categorizing, and forming generalizations about the phenomenon. “*Cognitive learning*” refers to the acquisition of skill and knowledge by cognitive or mental processes. “*Cognitive processes*” involve creating mental symbols of events and physical objects, and other methods used to process information.

Studies now show that “*cognitive learning*” is the main determinant with regards to evaluating a person’s learning ability with the following advantages: it improves learning, enhances concentration and perception, and facilitates the logical relationship between reasoning and facts. This type of learning led to “*comprehension,*” allowing an understanding of the topic and how to fit in other elements; “*memory*” allowing storing methods and the recall of them; and “*applications developing problem-solving skills for new situations.*”

7.6.1 Cognitive learning and reasoning

There many forms of “*Cognitive Learning and Reasoning*” but we focus mainly on some forms of “*learning through logic reasoning*” to apply “*AI models*” under “*Cognitive Computing.*” These can be separated generally into “*deductive reasoning,*” “*inductive reasoning,*” “*abductive reasoning,*” “*metaphoric reasoning,*” “*neuro-fuzzy logic reasoning,*” “*visuospatial relational reasoning,*” and others, as shown in Fig. 7.5 and explained in the next sections.

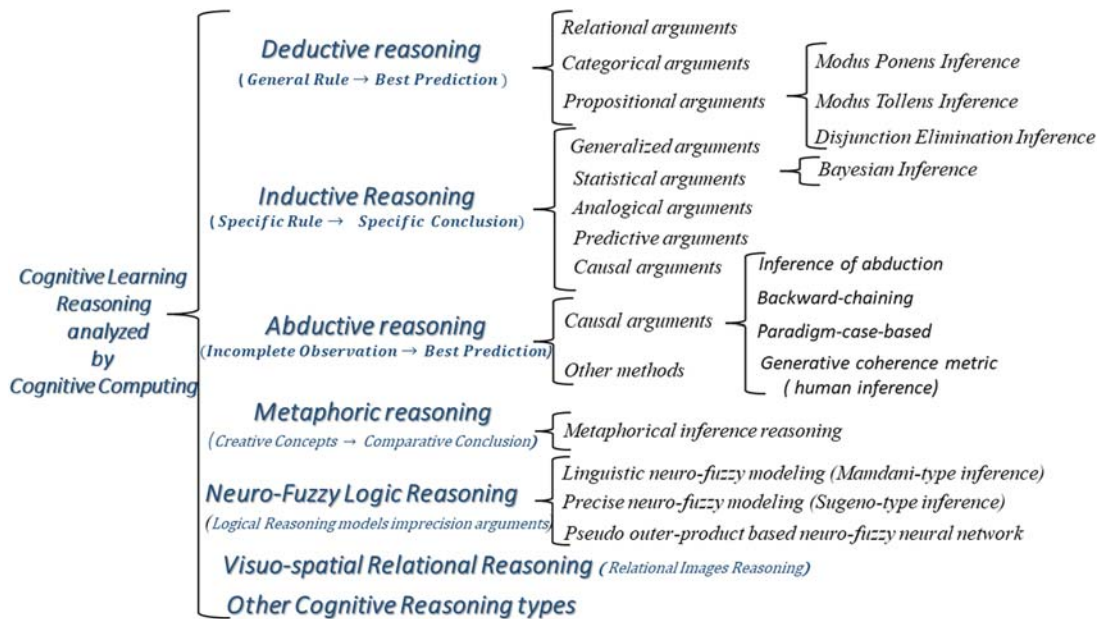


FIGURE 7.5 Cognitive learning reasoning analyzed by cognitive computing through different types of reasoning.

7.6.2 Deductive reasoning

“Deductive reasoning” is the process of drawing conclusions that are guaranteed to follow from given premises. There are many types of “deductive reasoning,” they are an essential element of “cognitive development and human thinking” with a focus on “deductive tasks, as the relational arguments” can be categorized into three classes [17]: “relational,” “categorical,” and “propositional,” where:

- “Relational” arguments are “A > B,” “B > C,” “Therefore, A > C”
- “Categorical” arguments are “All As = Bs,” “All Cs = Cs.” “Therefore, all As = Cs”
- “Propositional” arguments are “If there is an A, then is a B.” “There is an A,” “therefore, there is a B.” “Propositional relational arguments” can be basically analyzed as inferences: “modus ponens,” “modus tollens,” and “disjunction elimination.”

7.6.2.1 “Deductive reasoning—Propositional Arguments—Modus Ponens”

“Modus Ponens” is the logic rule for accepting the consequences, stating: “if a conditional statement is accepted, and the antecedent holds, then the consequences may be inferred”; it can be summarized as: “if A then B,” “A,” “therefore, B.” This rule can be represented as an equation, as shown in Eq. (7.3), and its inference rule in AI is “A implies B and A is true, then B is true,” as shown in Truth Table 7.1.

$$\text{Deductive reasoning —} \\ \text{Modus ponens } (A \rightarrow B), A \therefore B \quad (7.3)$$

Truth Table 7.1 Deductive reasoning—Modus ponens

A	B	A → B	A ∴ B
0	0	0	0
0	1	1	0
1	0	0	0
1	1	1	1

7.6.2.2 “Deductive reasoning—Propositional Arguments—Modus Tollens”

“Modus Tollens” is the logic rule for denying the consequences, stating: “if a conditional statement is accepted, and the antecedent is negated, then the consequences may be inferred as the negation”; it can be summarized as: “if A then B,” “not A,” “therefore, not B.” This rule can be represented as an equation, as shown at Eq. (7.4), and its

inference rule in AI is “A implies B and A is false, then B is false,” as shown in Truth Table 7.2.

$$\text{Deductive reasoning — Modus tollens } (A \rightarrow B), \bar{A} \therefore \bar{B} \quad (7.4)$$

Truth Table 7.2 Deductive reasoning—Modus tollens

A	B	\bar{A}	\bar{B}	A → B	$\bar{A} \therefore \bar{B}$
0	0	1	1	1	1
0	1	1	0	1	0
1	0	0	1	0	0
1	1	0	0	1	0

7.6.2.3 “Deductive reasoning—Propositional Arguments—Disjunction Elimination”

“Disjunction Elimination” is the valid argument form and rule of inference that allows one to eliminate a disjunctive statement, stating: “if conditional statement 1 or statement 2, therefore if statement 1 is false therefore statement 2 is true. It can be summarized as “if A or B, then if not A therefore B.” This rule can be represented in Eq. (7.3), and its inference rule in AI is A or B and A is false, then B is true,” as shown in Truth Table 7.3.

$$\text{Deductive reasoning —} \\ \text{Disjunction elimination } (A \cup B) \rightarrow A \therefore B \quad (7.5)$$

Truth Table 7.3 Deductive reasoning—Disjunction elimination

A	B	A ∪ B	\bar{A}	$\bar{A} \therefore B$
0	0	0	1	0
0	1	1	1	1
1	0	1	0	0
1	1	1	0	0

7.6.3 Inductive reasoning

“Inductive reasoning” is when the premises show evidence for a general possible truth. “Inductive reasoning” is the opposite of “deductive reasoning.” “Inductive reasoning” makes broad generalizations from specific observations, based on actual data to obtain a conclusion that can be true or false [18].

“Psychological research” has found that “inductive reasoning tests” are highly valid measures of a “person’s cognitive ability.” Some types of “inductive reasoning” have the following kinds of arguments: “Generalized,”

“Statistical,” “Bayesian,” “Analogical,” “Predictive,” and “Causal Inference,” where:

7.6.3.1 “Inductive reasoning—Generalized arguments”

“Inductive reasoning—Generalized arguments” is when a small sample of data makes a generalization about the whole population, that is, “in my town everybody uses right-hand driving; therefore, all in the United States use right-hand driving.”

7.6.3.2 “Inductive reasoning—Statistical arguments”

“Inductive reasoning—Statistical arguments” is similar to “generalized inductive reasoning” but uses a small set of statistics to make a generalization, that is, “Since 90% of the people brush their teeth in my town; therefore 90% of the people in the world brush their teeth.” “Bayesian inference” is a method of “statistical inference” using aspects of the scientific method, which involves collecting evidence that is meant to be consistent or inconsistent with a given hypothesis. As evidence accumulates, the “degree of belief” in a hypothesis ought to change. It assumes in the beginning a prior probability for a hypothesis based on logic or previous experience and, when faced with evidence, it adjusts the strength of the belief in that hypothesis in a precise manner using “Bayesian logic.” With enough evidence, it should become “very high or very low.” “Bayesian inference” computes the posterior probability according to “Bayes’ theorem,” as shown in Eq. (7.6).

$$\text{Bayes' theorem } P(H|E) = \frac{P(E|H) * P(H)}{P(E)} \quad (7.6)$$

where

H hypothesis, $P(H)$ is the prior probability, E is the current evidence,

$P(H|E)$ is the posterior probability of H given E after E is observed and $P(E|H)$ is the probability likelihood of observing E given H , $P(E)$ is the marginal likelihood.

7.6.3.3 “Inductive reasoning—Analogical arguments”

“Inductive reasoning—Analogical arguments” is when a hypothesis is drawn by analyzing two or more similar premises and their similarities, that is, stating “if A and B and C are similar in properties, then if A is similar in properties to D then it is possible that B and C are similar in properties to D .” “Analogies” enable a form of “inductive reasoning” that allows us to learn about a new situation based on prior knowledge about a similar

situation. This rule can be represented as an equation, as shown at Eq. (7.7) and its inference rule in AI is “ A and B and C are similar then (if A similar D then B similar D and C similar D),” as shown in Truth Table 7.4

$$\text{Inductive reasoning — Analogical} \quad (7.7)$$

$$((\sim A) \rightarrow (\sim B) \rightarrow (\sim C)) \text{ then } ((\sim A) \rightarrow (\sim D)) \\ \tilde{\rightarrow} ((\sim B) \rightarrow (\sim D)) \cap ((\sim C) \rightarrow (\sim D))$$

where \sim represents similar to, and $\tilde{\rightarrow}$ represents possible to.

Truth Table 7.4 Inductive reasoning—Analogical

A	B	C	D	$(\sim A) \rightarrow (\sim D)$	$(\sim B) \rightarrow (\sim D)$	$(\sim C) \rightarrow (\sim D)$
0	0	0	0	0	0	0
1	1	1	1	1	1	1

7.6.3.4 “Inductive reasoning—Predictive arguments”

“Inductive reasoning—Predictive arguments” is when a hypothesis is drawn about the future using data from the past, that is, “In the last years it has been raining in October. Therefore, this year it will rain in October.”

7.6.3.5 “Inductive reasoning—Causal arguments”

“Inductive reasoning—Causal arguments” is when an inductive logic draws a causal link between a premise and a hypothesis, it is quite commonly applied as “ A caused B ” or “ B occurred because of X ,” that is, “The lack of vigilance in this street causes one to get robbed.”

7.6.4 Abductive reasoning

“Abductive” typically begins with a possible incomplete set of observations, and proceeds to the likeliest possible explanation for the set, that is, “a medical diagnosis given this set of symptoms, where the diagnosis would best explain most of them.” The “abductive inductive reasoning” can be creative, intuitive, revolutionary, and typically “uncertain.” “Albert Einstein’s work” was done as a “thought experiment,” nevertheless, his deductions appear to have been right until now.

“Abductive reasoning” is the logical process where one chooses a hypothesis that would best fit the given facts. “Abductive conclusions” are thus qualified as having a remnant of uncertainty or doubt, which is expressed in terms, such as “best available” or “most likely.” “Abductive Reasoning” is used in many “AI application,” such as “Medical Diagnosis,” “Natural Language Understanding,” “Plan Understanding,” and “Scientific

Reasoning and discovery” that can be analyzed with “causal arguments” based on “a set of inferences (causes)” and a “set of facts (effects),” as represented in Eq. (7.8)

Abductive Reasoning — Causal arguments (cause \supset effect)
(7.8)

Given $(p(t_1), q(t_1), \dots, p(t_n), q(t_n)) \therefore (p(x) \supset q(x))$
where: p is sufficient for q

General steps for: “Abductive Reasoning—Causal arguments”

1. “(Cause \supset Effect)”
2. “Find one or more explanations of the observed facts in terms of knowledge and observations”
3. “Select one or more as the most likely cause(s)”
4. Then, “add more assets of inferences”
5. And repeat the cycle on time to fine tune the selection (s)”

Following the general steps for “Abductive Reasoning—Causal reasoning,” we can infer different approaches, such as “Inference of abduction,” “Backward-chaining,” “Paradigm case-based,” “Generative metric,” and others.

7.6.4.1 “Abductive reasoning—Causal arguments—Inference of abduction”

“Inference of abduction” is an approach that reveals where errors can and do occur and how such errors might be reduced or even eliminated. It is a kind of “Abductive Reasoning for causal reasoning” that can be used for “inferential reasoning driving clinical diagnosis,” that often takes place subconsciously, and so rapidly that its nature remains largely hidden from the diagnostician. Nevertheless, some researchers [19] have proposed that raising such reasoning to the conscious level reveals a “clinical diagnostic reasoning in terms of a pattern of IF/THEN/THEREFORE reasoning driven by data gathering and the inference of abduction.”

7.6.4.1.1 Example: “Inductive reasoning—Causal arguments—Inference of abduction”

Problem to resolve

A patient goes to a doctor because she has abdominal pain, the doctor after an abdomen checkup has the hypothesis that she has “Cholecystitis (inflammation of gallbladder).” Apply “Inductive reasoning—Causal arguments—Inference of abduction” to confirm the hypothesis

Procedure

The doctor could apply the “inference of abduction” approach using “IF/THEN/THEREFORE reasoning”

based on the “Current knowledge storage AI Database (KSAIDB)” as follows:

1. Current knowledge storage AI Database (KSAIDB) with information as (Cause \supset Effect)
 - a. (Cholecystitis symptoms \supset pain in upper right abdomen, pain travels to right shoulder or back, abdominal tenderness, nausea, vomiting)
 - b. (Cholecystitis causes \supset Gallstone blocking exit tube of gallbladder, bile duct problems, tumors, infection of gallbladder, trauma or injury by accidents, obesity, high-fat diet)
 - c. (Cholecystitis diagnosis \supset complete blood count, bilirubin test, ultrasound, CT scan, hepatobiliary iminodiacetic acid scan)
2. IF/THEN/THEREFORE reasoning
 - a. IF Cholecystitis is the cause for her abdominal pain, THEN he could order a blood test, THEREFORE the results must show increased levels of white blood cells indicating and elevated liver test to be an infection
 - b. IF the blood test results show high levels of white blood and slightly elevated value “livers tests,” THEN could be a “gallstone,” THEREFORE the doctor could order an “abdomen ultrasound”
 - c. IF the “abdomen ultrasound” shows “inflammation of the gallbladder” as an accumulation of bile due to “gallstone blockage in the exit tube (cystic duct),” THEN the hypothesis of a gallstone would have some support, THEREFORE he could decide for a prescription of a medication to dissolve the gallstone, and an analgesic to manage the pain
 - d. The doctor could request to see her in 15 days or before to check if she does show any improvement in her abdominal pain in the belly

7.6.4.2 “Abductive reasoning—Causal arguments—Backward-chaining”

“Abductive reasoning—Causal arguments—Backward-chaining” is a repetitive approach to find proof of an original fact that follows the four next steps [20] <http://cogsys.org/app/webroot/courses/langley/aicogsys11/notes/abduct.pdf>:

1. “Fact that needs to be explained”
2. “The fact is chained through a rule”
3. “Unify the rule antecedents with fact as possible”
 - a. “Chain off unmatched antecedents when not possible”
 - b. “Make default assumptions for unmatched antecedents”
4. Process continues until producing a “proof of the original fact that terminate in other facts or assumptions*”

Note*: This is a “Query-Driven” using “AND-OR search” through the space of explanations.

7.6.4.2.1 Example “Abductive reasoning—Causal arguments—Backward-chaining”

Problem to resolve

Fact: “John is healthy, he usually eats and sleep well.” Apply “Abductive reasoning—Causal arguments—Backward-chaining” to reasoning for this fact.

Procedure

Use a repetitive approach to find proof of an original fact using the four steps for “Inductive reasoning—Backward-chaining.”

1. “Fact that needs to be explained,”
 - a. Fact: (John is healthy, he usually exercises, eats, and sleeps well)
2. “The fact is chained through a rule”
 - a. Rules: (Healthy \supset eat well), (Healthy \supset sleep well), (Healthy \supset exercise regularly)
3. “Unify the rule antecedents with facts as possible”
 - a. Unify: (Healthy \supset eat well, sleep well, exercise regularly)
4. Process continues until producing a “proof of the original fact that terminates in other facts or assumptions”
 - a. Proof: (John is healthy because he exercises, eats well, and sleeps well)

7.6.4.3 “Abductive reasoning—Causal arguments—Paradigm case-base”

“Abductive reasoning—Causal arguments—Paradigm case-base” is an approach that uses cases representation to produce explanation of anomalies that follows the next four steps:

1. “Encodes knowledge as cases-based representation” instead of general rules
2. “Generates accounts that do not correspond to proof trees”
3. “Attempts to find explanation only when anomalies occur,”
4. “Produce explanation by adapting cases that fail to handle these anomalies”

7.6.4.3.1 Example: “Abductive reasoning —Causal arguments—Paradigm case-based”

Problem to resolve

“Specify abnormal body temperatures.” Apply “Abductive reasoning—Causal arguments—Paradigm case-based” to explain abnormal body temperatures.

Procedure

Apply cases representation to produce explanation of anomalies, following the four steps for “Abductive reasoning—Paradigm case-based”:

1. “Encodes knowledge as cases-based representation”
 - a. (Normal body temperature, 98.6°F),

- b. (Normal body temperature, above than 97°F and below than 99°F)
2. “Generates accounts that do not correspond to proof trees”
 - a. (Abnormal body temperature, 97°F or below)
 - b. (Abnormal body temperature, 99°F or above)
 - c. (Abnormal body temperature is hypothermia, below than 95°F)
 - d. (Abnormal body temperature is fever, above than 99°F)
3. “Attempts to find explanation only when anomalies occur,”
 - a. (if body temperature is 94°F then is Abnormal)
 - b. (if body temperature is 100°F then is Abnormal)
4. “Produce explanation by adapting cases that fail to handle these anomalies”
 - a. (if body temperature is 94°F then is Abnormal and has hypothermia)
 - b. (if body temperature is 101°F then is Abnormal and has fever)

7.6.4.4 “Abductive reasoning—Causal arguments—Generative coherence metric (Human inference)”

“Abductive reasoning—Causal arguments—Generative coherence metric” is an approach that follows a typical “human inference” applying the four next steps:

1. “Accept new facts as they arrive”
2. “Form explanations in an incremental way, based on Current Knowledge”
3. “Continually execute backward and forward over its rules”
4. “Calculate coherence metric, ensuring that inference will scale”

7.6.4.4.1 Example: “Abductive Reasoning—Causal arguments—Generative Coherence metric”

Problem to resolve

Conclude if a patient “x” and a patient “y” has: “flu” or “COVID-19.” Apply “Abductive reasoning—Causal arguments—Generative coherence metric” to reach the conclusion.

Procedure

“Abductive reasoning—Causal arguments—Generative coherence metric” uses an approach that follows a typical “human inference,” applying the four steps of “Abductive reasoning—Causal arguments—Generative coherence metric” [21]:

1. “Accept new facts as they arrive”

New facts:

 - a. (person x has fever, chills, headache, cough)

- b. (person y has fever, chills, headache, diarrhea, loss of smell)

“Knowledge storage AI Database with information as (Cause \supset Effect)

- a. (Flu symptoms \supset fever, chills, headache, cough, muscle pain, fatigue, nausea, diarrhea)
 b. (COVID-19 symptoms \supset fever, chills, headache, cough, muscle pain, fatigue, nausea, diarrhea, shortness of breath, sore throat, loss of taste, loss of smell)
 c. (Flu and COVID-19 have symptoms in common)
 d. (Flu and COVID-19 have some similar symptoms)
2. “Form explanations in an incremental way, based on Current Knowledge Storage AI Database (KSAID)”
- “Form explanations in incremental way”
- a. (person x probably has flu, has flu symptoms)
 b. (person y probably has COVID-19, has some flu symptoms and COVID-19 symptoms)

“Conclusion”

- a. (person x is requested for a COVID-19 test)
 b. (person y is requested for a COVID-19 test)
 3. “Continually execute backward and forward over its rules”

“Update Knowledge Storage AI Database”

- a. (person x \supset COVID-19 test as negative)
 b. (person y \supset COVID-19 test as positive)

“Update Conclusion”

- a. (person x has probably flu symptoms)
 b. (person y has probably COVID-19 symptoms)
 4. “Calculate coherence metric, ensuring that inference will scale”

“Calculate coherence metric”

- a. (person x has flu symptoms, “follow flu procedures”)
 b. (person y has flu symptoms, “follow COVID-19 procedures”)

“Follow patient medical history”

- a. (person x must follow flu diagnosis and treatment)
 b. (person y must follow COVID-19 diagnosis and treatments)

7.6.5 Abductive reasoning for medical diagnosis

Methods, such as “Abductive Reasoning—Causal arguments (cause \supset effect)*,” as shown in Eq. (7.8), are based on a simple structure that can be useful for many injuries, illnesses, and diseases.

Note*: Some diseases as the “neurologic diseases” are not easy to diagnose, and there are not actually specific tests to diagnose them. The doctor trained in

nervous system conditions, a “neurologist,” will diagnose them based on the medical history, a review of signs and symptoms, and a neurological and physical examination [22].

7.6.5.1 Example: Abductive reasoning for medical diagnosis

Problem to resolve

Conclude if a patient “x” has “Parkinson’s disease.” Apply “Abductive reasoning” to conclude this “Medical Diagnosis.”

Procedure

Actually, the only way to diagnose a neurological disease like “Parkinson’s” is based on medical history, a review of the signs and symptoms, and a neurological and physical examination. There is not an easy way to analyze the many complex issues deriving from this disease:

- The exact cause of functional neurologic disorders is unknown.
- Symptoms are complex and involve multiple mechanisms that may differ, depending on the type of functional neurologic disorder.
- Areas of the brain that control the functioning of your muscles and senses may be involved, even though no disease or abnormality exists.
- Symptoms of functional neurologic disorders may appear suddenly after a stressful event, or with emotional or physical trauma.
- Triggers may include changes or disruptions in how the brain functions at the structural, cellular, or metabolic level. But the trigger for symptoms cannot always be identified.

It is probable that Doctors may suggest a specific “single-photon emission computerized tomography (SPECT) scan” called a “dopamine transporter scan (DaTscan*).” Although this can help support the suspicion of “Parkinson’s disease,” it is the symptoms and neurological examination that ultimately determines the correct diagnosis.

Note*: Most people do not require a “DaTscan.”

Using these types of reasoning in complex problems will help someday, but they must evolve and grow based on more new research findings, new experiences, new treatments, new tests, new finding in scanning images, etc., to help in the development of this kind of complex “AI Models.” It will also require more complex analysis, such as “DNA” composition and related tests, images, etc., and probably will need powerful hardware and complex algorithms to analyze them.

7.6.6 Metaphoric reasoning

“*Metaphoric reasoning*” is defined as the cognitive act that enables a description of an object or event, real or imagined, using concepts that cannot be applied to the object or event in a conventional way [23]. Typically, “*Metaphorical inference reasoning*” is applied when a description of an object or event, real or imagined, is based on using creative design concepts to explain a conclusion [24]. For “*applied Biomedical Engineering using artificial intelligence and cognitive models*” a “*Metaphorical inference reasoning*” is a linguistic manifestation of a “*cognitive process*” based on “*analogical reasoning*,” and usually consists of three parts, as indicating in Eq. (7.9) [25]: “*target*,” “*source*,” and “*analogical mapping*.”

$$\text{Metaphoric reasoning } \{target\} \text{ LIKE } \{source\} \text{ THAT } \{analogical \text{ mapping}\} \quad (7.9)$$

“*Metaphoric reasoning*” using “*Metaphorical inference reasoning*” based on “*analogical reasoning*” is a method of acquiring new information by the inspection of a specific instance to obtain similarities, that can be built by four stage process: “*recalling*,” “*mapping*,” “*testing*,” and “*recurrence*”:

1. “*Recalling*” one or most past problems that have strong similarity to the new problem.
2. “*Mapping*” from the old problem solution process into a solution for the new problem, based on known similarities between both.
3. “*Testing*” the potential known similarities and redefining it, if necessary.
4. “*Recurrence*” of a solution pattern problem as a “*reusable rule*” for a common type of problem.

7.6.6.1 Example: metaphoric reasoning

Metaphor: “*COVID-19 is LIKE a cancer THAT kills mostly with the overreaction of the immune system.*” Apply *Metaphoric reasoning* to explain this metaphor, where:

- “*Target*” is “*COVID-19*”;
- “*Source*” is “*cancer*”; and
- “*Analogical mapping*” is “*kills mostly with the overreaction of the immune system.*”

A “*Metaphoric reasoning*” example using “*Metaphorical inference reasoning*” based on “*analogical reasoning*,” applying (cause \supset effect) in “*AI and Cognitive Computing Agent System with Artificial Continue Intelligence framework*” is shown in Fig. 7.4. The similarities between “*Cancer*” and “*COVID-19*” are built in two parts: “*Current Knowledge Storage AI Database (KSAIDB)*” and the “*Metaphorical inference reasoning*” four-stage process.

1. Current Knowledge Storage AI Database (KSAIDB) [26]

a. KSAIDB: Immune blood cell

- i. (*immune system \supset complex network of cells, tissues, and organs that helps the body fight infections and other diseases*)
- ii. (*immune system function \supset fights infections using white blood cells*)
- iii. (*blood cell \supset red blood cells, white blood cells, platelets, and plasma*)
- iv. (*White blood cells \supset leukocytes*)
- v. (*leukocytes \supset born in bone marrow cells*)
- vi. (*leukocytes \supset protects against illness and disease*)
- vii. (*leukocytes \supset flow in the bloodstream to fight viruses, bacteria, and other foreign invaders*)

b. KSAIDB: Cancer

- i. (*cancer types \supset change way immune blood cells work*)
- ii. (*cancer cells \supset get into bone marrow cells, compete with leukocytes*)
- iii. (*cancer types \supset lymphomas, multiple myeloma, leukemia, others*)
- iv. (*lymphomas cancer attack \supset change way immune blood cells work*)
- v. (*multiple myeloma cancer attack \supset change way immune blood cells work*)
- vi. (*most types of leukemia attack \supset change way immune blood cells work*)

c. KSAIDB: COVID-19 [26]

- i. (*COVID-19 disease \supset coronavirus SARS-Cov-2*)
 - ii. (*coronavirus types \supset change way immune blood cell work*)
- d. (*coronavirus steps \supset infect airways, multiply inside cells, overreact immune system function, others*)
- e. (*coronavirus overreact immune system function \supset over-activate leukocytes produce cytokine storm*)
- f. (*coronavirus cytokine storm \supset release great amounts of cytokines produce inflammation stimulating molecules into the blood*)
- g. (*coronavirus lungs \supset cytokine storm attracts excess of immune cells into lung tissue causing lung injury*)

2. “*Metaphorical inference reasoning*” four stage process

- a. “*Recalling*” one or most past problems that have strong similarity to the new problem
 - i. (*cancer types \supset change way immune blood cell work*)
- b. “*Mapping*” from the old problem solution process into a solution for the new problem, references the know similarities between both

- i. (coronavirus types \supset change way immune blood cell work)
- c. “Testing” and redefining the potential known similarities
 - i. “COVID-19 is LIKE a cancer”
- (cancer cells \supset get into bone narrow cells, compete with leukocytes)
- (leukocytes \supset flow bloodstream to fight viruses, bacteria, and other foreign invaders)
- (coronaviruses overreact immune system function \supset over-activate leukocytes produce cytokine storm)
- (coronavirus cytokine storm \supset release great amounts of cytokines produce inflammation stimulating molecules into the blood)
- d. “Recurrence” of a solution pattern problem as a “reusable rule,” for common type of problems
 - i. Rule: “COVID-19 is LIKE a cancer THAT kill mostly with overreaction of the immune system”
 - ii. Reusable rule: “{target} LIKE {source} THAT {analogical mapping}”

and “NO.” It was studied in my last book “Applied Biomechatronics Using Mathematical Models” in Chapters 6 [27] and 7 [10]. The “Fuzzy logic reasoning” generally reaches a conclusion following seven steps:

1. “Define linguistic variables for input and output”;
2. “Construct knowledge base rules based on experiences”;
3. “Define fuzzy rules to relate the variables”;
4. “Fuzzification, which consists of converting crisp data into fuzzy sets using the membership functions”;
5. “Evaluate rules in the Fuzzy Inference System (FIS)”
6. “Combine results for each rule by FIS”;
7. “Defuzzification, which converts the output data into nonfuzzy values.”

“Fuzzy logic reasoning” is based on two types of “Inferences Fuzzy Systems,” which differ in the way that the outputs are determined. These are: “Mamdani-type inference” and “Sugeno-type inference.”

7.6.7 Neuro-Fuzzy logic reasoning as cognitive reasoning

“Neuro-Fuzzy Logic Reasoning” is based on “Fuzzy logic reasoning,” which “resembles human reasoning in a similar to way how humans perform decision-making,” and it involves all intermediate possibilities between “YES”

- “Mamdani-type inference” is well-suited to human input, uses an intuitive and interpretable rules base, thus allowing an easy understanding. It is frequently created from human expert knowledge, such as “medical diagnostics” as shown in the next example and represented in Fig. 7.6

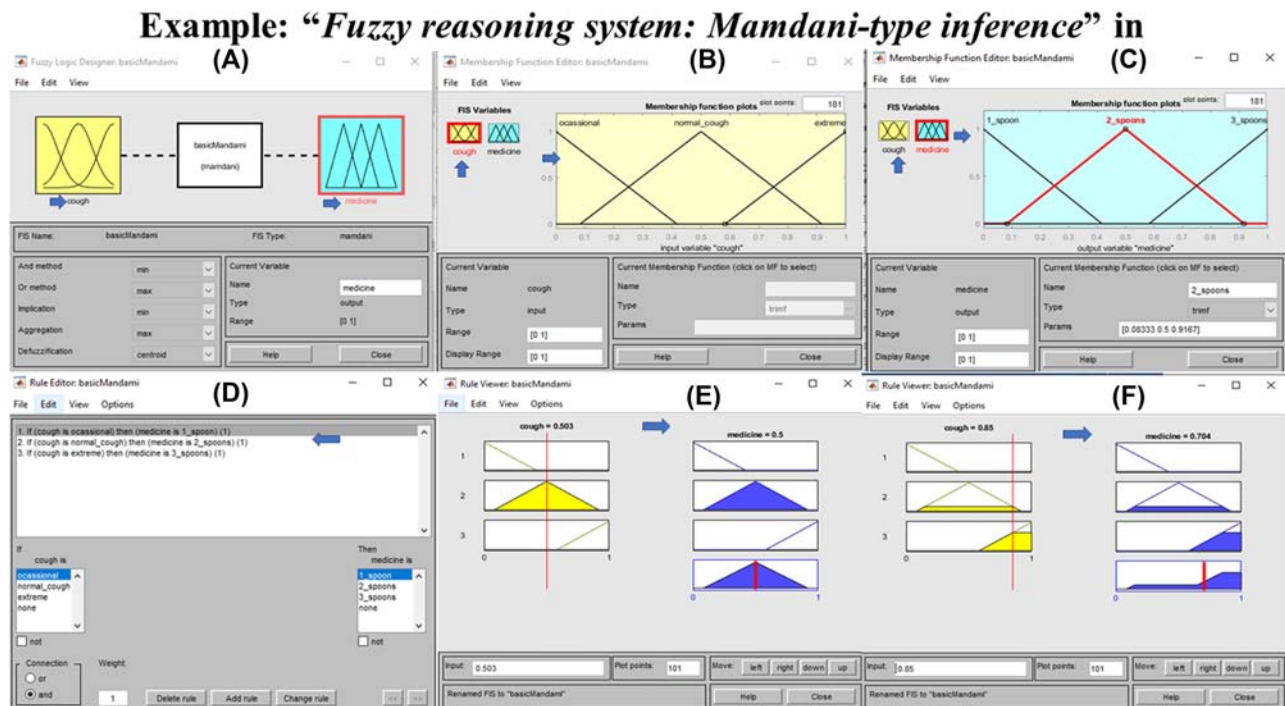


FIGURE 7.6 Basic example of “Fuzzy logic reasoning” based on “Mamdani-type inference.”

7.6.7.1 Example “Fuzzy System: Mamdani-type inference” in MATLAB

Problem to resolve:

Assume a “liquid medicine for control cough” and analysis is required using “Fuzzy Inference systems to control cough.” Apply “Mamdani-type inference” to build a “Inference Fuzzy System.”

Procedure:

In MATLAB in the command prompt enter “fuzzy,” which loads the “Fuzzy Logic Designer*,” and in the menu select: *File > Import > From File* “basicMamdani.fis**”

Note*: The Fuzzy Logic Toolbox is required

Note**: This file can be found in the companion directory of this book, at the following directory: “... \MATLAB\Exercises_book_ABME\CH7\MATLAB NEURO-FUZZY”

Follow the seven steps as:

Step (1) “Define linguistic variables for input and output”

- Input variable = “cough” and output variable = “liquid medicine,” as shown in Fig. 7.6A

Step (2) “Construct knowledge base rules based on experiences”

- “One spoon of liquid medicine every 8 hours” is recommended for “occasional cough”
- “Two spoons of liquid medicine every 8 hours” is recommended for “normal cough”
- “Three spoons of liquid medicine every 8 hours” is recommended for “extreme cough”

Step (3) “Define fuzzy rules to relate the variables”

- Rule 1: If (cough is occasional) then (medicine is 1_spoon)
- Rule 2: If (cough is normal) then (medicine is 2_spoons)
- Rule 3: If (cough is extreme) then (medicine is 3_spoons)

Step (4) “Fuzzification, which consists of converting crisp data into fuzzy sets using the membership functions”

- Input Variable = “cough, as shown in Fig. 7.6B
- Output Variable = “liquid medicine,” as shown in Fig. 7.6C

Step (5) “Evaluate rules in the Fuzzy Inference System (FIS)”

- If input “cough = 0.503,” then output “medicine = 0.5.” The fuzzy rules are shown in Fig. 7.6D and their evaluation in Fig. 7.6E.

Step (6) “Combine results for each rule by FIS”

- If input “cough = 0.85,” then output “medicine = 0.704.” As shown in Fig. 7.6F.

Step (7) “Defuzzification, that convert the output data into nonfuzzy values”

- Fuzzy input “medicine = 0.503” is crispy value is 2 spoons
- Fuzzy output “medicine = 0.704” is crispy value is approx. 2.112 spoons

The “Sugeno-type inference” is computationally efficient, works well with optimization and adaptive techniques, and it is frequently used in mathematical analysis. “Sugeno-type inference,” also known as “Takagi-Sugeno-Kang fuzzy inference,” uses “singleton output membership functions” that are either constant or a linear function of the input values. The “defuzzification” process for a “Sugeno system” is more computationally efficient compared to that of a “Mamdani system”; it uses a weighted average or weighted sum of a few data points rather than computes a “centroid” of a two-dimensional area. This “Sugeno-type inference” is frequently used to act as an interpolating supervisor of multiple linear controllers that are to be applied, respectively, to different operating conditions of a dynamic nonlinear system, as shown in the example in Fig. 7.7.

7.6.7.2 Example “Fuzzy System: Sugeno-type inference” in MATLAB

Problem to resolve

Assume again the same problem to resolve from the last example: “liquid medicine for control cough” and analysis required using “Fuzzy Inference systems based on the Sugeno-type inference” to control “cough.”

Procedure

In MATLAB in the command prompt enter “fuzzy,” which loads the “Fuzzy Logic Designer*,” and in the menu select: *File > Import > From File* “basicSugeno.fis**”

Note*: The Fuzzy Logic Toolbox is required

Note**: This MATLAB variable can be found at the companion directory of this book, at the following directory: “... MATLAB\Exercises_book_ABME\CH7\MATLAB NEURO-FUZZY”

Follow the seven steps:

Step (1) “Define linguistic variables for input and output”

- Input variable = “cough” and output variable = “liquid medicine,” as shown in Fig. 7.7A.

Step (2) “Construct knowledge base rules based on experiences”

- “One spoon of liquid medicine every 8 hrs” is recommend for “occasional cough”
- “Two spoons of liquid medicine every 8 hrs” is recommend for “normal cough”
- “Three spoons of liquid medicine every 8 hrs” is recommend for “extreme cough”

Example: “Fuzzy reasoning system: Sugeno-type inference” in MATLAB

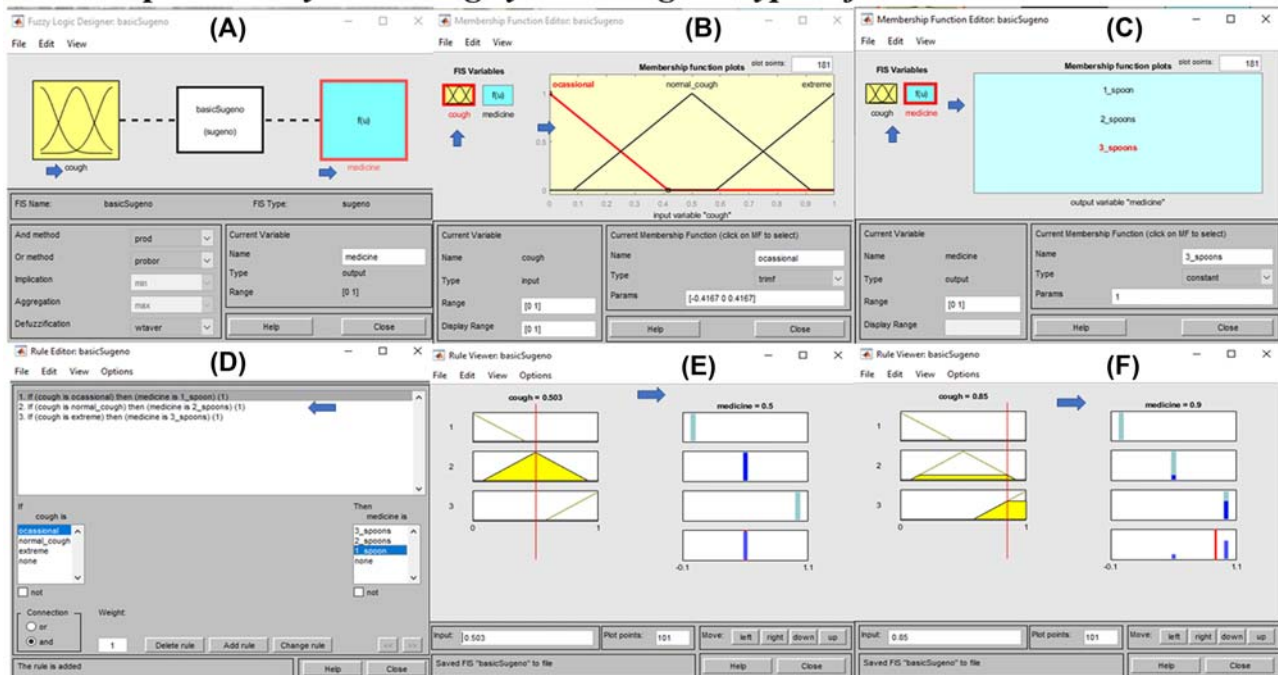


FIGURE 7.7 Basic example of “Fuzzy logic reasoning” based on “Sugeno-type inference.”

Step (3) “Define fuzzy rules to relate the variables”

- Rule 1: If (cough is occasional) then (medicine is 1_spoon)
- Rule 2: If (cough is normal) then (medicine is 2_spoons)
- Rule 3: If (cough is extreme) then (medicine is 3_spoons)

Step (4) “Fuzzification, that consist in convert crisp data into fuzzy set using the membership functions”

- Input Variable = “cough, as shown in Fig. 7.7B
- Output Variable = “liquid medicine,” as shown in Fig. 7.7C. Verify that each membership has a parameter: “1_spoon = 0,” “2_spoons = .5,” and “3_spoons = 1”

Step (5) “Evaluate rules in the Fuzzy Inference System (FIS)”

- If input “cough = 0.503,” then output “medicine = 0.5.” The rules to evaluate are shown in Fig. 7.7D and the evaluation in Fig. 7.7E

Step (6) “Combine results for each rule by FIS”

- If input “cough = 0.85,” then output “medicine = 0.9*.” As shown in Fig. 7.7F

Note*: This value in “Sugeno-type inference” is more precise than that obtained using the “Mamdani-type inference.”

Step (7) “Defuzzification, that convert the output data into nonfuzzy values”

- Fuzzy input “medicine = 0.503” is crispy value = 2 spoons
- Fuzzy output “medicine = 0.9” is crispy value = 2.7 spoons

“Neuro-Fuzzy systems (NFS)” to be studied in the next section, is the integration of “Fuzzy systems,” such as the “human-like reasoning of fuzzy systems,” with “Artificial Intelligence,” such as “machine learning (ML),” “artificial neural network (ANN),” and “cognitive computing models.” The typical steps for a “NFS” applied for “Medical diagnostic processes” are usually based on signs, symptoms, and laboratory investigations following the steps: “integrate a dataset for medical diagnosis,” “partition the original dataset in training and validation dataset,” “define Fuzzy Inference structure type,” “obtain a FIS rules applying AI algorithms,” “optimize the FIS to improve precision in the FIS rules applying AI optimization algorithms,” “apply FIS algorithm to analyze new patient data,” and “defuzzification”:

- “Integrate a dataset for medical diagnosis,” where the dataset includes all variables needed to describe the characteristics of the objective to analyze based on trustable historical differential symptoms systems of many patients that have experienced the objective,

graded by values between “normal = 0” through to “abnormal = 1.”

- “Partition the original dataset in training and validation dataset.” The original complete “dataset for each specific medical diagnosis” is split in equal sizes into “training dataset” and “validation dataset.”
- “Define Fuzzy Inference structure type” specifying its inputs and output variables, where each variable represents each of the data attributes and ranges from “0” to “1.” It is important to reduce the number of possible “fuzzy rules,” that is, using two “membership function (MFs)” for each input variable, will result in $2^9 = 512$ MFs for the output variable for best precision or using $512/2 = 256$, $512/4 = 128$, or $512/8 = 64$ losing precision for faster results.
- “Obtain a FIS rules applying AI algorithms” where the training can be made by applying a different algorithm such as “genetic algorithm,” “particle swarm,” “simulated annealing algorithm,” “adaptive neuro-fuzzy,” and others, to generate the rules specified in the output.
- “Optimize the FIS to improve precision in the FIS rules applying AI optimization algorithms,” this optimization is also known as “tuning,” and applying algorithms such as “genetic algorithm,” “particle swarm,” “pattern search,” “simulated annealing algorithm,” “adaptive neuro-fuzzy,” and others, to simplify the number of rules obtained during the training.
- “Apply FIS algorithm to analyze new patient data.” The FIS model is applied to predict new data to be classified in the output based on the optimized fuzzy rules
- “Defuzzification, which converts the output data into nonfuzzy values.”

7.6.7.3 “Neuro-Fuzzy systems (NFS)”

“Neuro-Fuzzy systems (NFS)” types can be divided basically into “linguistic fuzzy modeling,” “precise fuzzy modeling,” and “pseudo outer-product based fuzzy neural network”:

- “Linguistic fuzzy modeling” based on interpretability applying the “Fuzzy System: Mamdani-type inference.”
- “Precise fuzzy modeling” is based on accuracy applying the “Fuzzy System: Sugeno-type inference.”
- “Pseudo Outer-product Based Fuzzy Neural Network” are “neuro-fuzzy systems” using in combination with other methods, such as “Approximate Analogical Reasoning Scheme” [28], “commonly accepted fuzzy Compositional Rule of Inference” [29], “self-organization and associative memories” [30], and others.

In summary, “Fuzzy logic reasoning” resembles human reasoning in the similar way how humans perform decision making, and it involves all intermediate possibilities between “YES” and “NO.” Its application for “Medical diseases” is a challenging task due to the nature of data, that can be incomplete, uncertain, and imprecise. But its exponential evolution is improving results with better precision each time, as shown in the example: Research 7.5, section 7.7.3 “Linguistic Neuro-Fuzzing Modeling to predict breast cancer tumor.”

7.6.8 Visuospatial relational reasoning

“Visuospatial relational reasoning” is a kind of “Relational reasoning” using images for reasoning. It investigates the relational reasoning using visuospatial materials with functional magnetic resonance imaging.

Based on our interest in “Neurology applying Artificial Intelligence to evaluate the human cognitive processes through cognitive computing,” and because of the analysis of learning, reasoning, and others, the correlation of biological processes with brain activity can be studied using imaging techniques, such as “fMRI” and other imaging bioinstruments, and by detecting neuropotential activities in the brain using “EEG.”

Many of the early neuroimaging studies of reasoning suggested that the “frontal lobes” in the “human brain” were particularly important. Later other studies indicated that the “frontal lobes” may serve a coordinating or control function, possibly integrating widescale activity across the brain, rather than operating as a relational module. “Brain network interconnectivity” looks to be an especially promising area toward capturing the complexity of “neural processing in reasoning,” and may further clarify some of the roles of specific brain areas that have been linked to reasoning in various forms. For example, the “prefrontal cortex (PFC)” tends to show advanced reasoning skills, and it is an active component of numerous reasoning tasks across several domains, including “visuospatial relational reasoning,” “analogical relational reasoning,” “deductive, and inductive reasoning” [31]. Besides, “deductive reasoning” has shown significant clusters of activation in the brain in special areas, such as “left inferior frontal gyrus (IFG),” “left medial frontal gyrus (MeFG),” “bilateral middle frontal gyrus (MFG),” “bilateral precentral gyrus (PG),” “bilateral posterior parietal cortex (PPC),” and “left Basal Ganglia (BG)” [32].

“Visuospatial relational reasoning” is a method that has indicated how the “human brain” is involved simultaneously in many activities at any moment in time. For example, when
(Continued)

(Continued)

typing on a keyboard, the brain sends the orders for the finger movements, but also is reasoning in how to form the words to describe the concept that is being documented, as well as undertaking other activities such as feeling tired, thirsty, etc. This is based on a concept known as “dynamic neural pathways” that standard “AI algorithms” are not analyzing. The new AI algorithms apply decoding of reasoning activities that will allow the analysis of complex brain activities, thus enabling new neuroscience discoveries and showing how can we get new “AI models” to analyze them [33].

7.6.9 Cognitive learning and relationship with neuroscience of reasoning

“Cognitive Learning and its relationship with the neuroscience of reasoning is proposed as Cognitive Learning-Reasoning (CL&R) using Cognitive Computing (CC).” “CL&R” can be defined in the following two steps:

- “Cognitive Learning” is a mental process that takes in, interprets, stores, and retrieves information to infer something, and
- “Cognitive Learning” can be studied by the “neuroscience of reasoning” that scientifically involves determining the neural correlates of reasoning.

These relationships can be investigated in many ways, such as “Visuospatial Relational Reasoning,” explained in Section 7.6.8, using the “event-related potential in the human brain by functional magnetic resonance imaging” [34], with the objective of obtaining “AI Models” applying “Cognitive Computing” methods.

In summary, “deductive reasoning,” “inductive reasoning,” “abductive,” and “metaphoric inference” directly refer to one thing by mentioning another. They may provide clarity or identify hidden similarities between two ideas based on mapping of a set of conceptualities that depend on each individual interpretation.

The different types of reasoning have the following characteristics [35,36]:

- “Deductive reasoning” is a top-down approach from premises to conclusion that guarantees the validity of a conclusion, provided that the premises are true. It can be represented as: (**General Rule** → **Best Prediction**).
- “Inductive Reasoning” is a bottom-up approach that does not guarantee a valid conclusion, only a probable conclusion based on some samples of the total population where inferences are drawn based on probability values. It can be represented as (**Specific Rule** → **Specific Conclusion**).

(Continued)

(Continued)

- “Abductive reasoning”: It can be represented as (**Incomplete Observation** → **Best Prediction**), maybe is true
- “Metaphoric reasoning” is a way of inferring conclusions having uncertainty or doubt expressed as most likely taking the best shot. It can be represented as (**Creative Concepts** → **Comparative Conclusion**).
- “Fuzzy Logic Reasoning” is intended to model logical reasoning with vague or imprecise statements, referring to the systematic handling of degrees of some kind for a partial true.
- “Visuospatial Relational Reasoning” is intended for relation reasoning based on images to obtain a best prediction.

7.7 Cognitive Learning and Reasoning research example applying AI-CCAS framework

In this section we will focus on examples of research applying different types of “Cognitive Learning and Reasoning using Cognitive Computing”:

- “Cognitive sentiments analysis for neurologic diseases that affect mood changes under Cognitive Computing.” The main objective is to obtain an “AI model” to classify the “mood change” for yesterday and today on “neurologic diseases, such as Parkinson’s disease (PD)”, and other diseases by analyzing their cognitive status using their mood descriptions using an “AI and Cognitive Computing Agents System (AICCAS) framework” using MATLAB. See Research 7.3, section 7.7.1 “Cognitive sentiments analysis for neurologic diseases that affect mood changes under Cognitive Computing.”
- “Deductive reasoning evaluation for neurologic diseases patients using Cognitive Computing.” The main objective is to obtain an “CNN-NLP model” to detect, analyze, and classify deductive reasoning evaluation texts of patients with neurologic diseases. It analyzes their cognitive status based on their answers for three types of deductive reasoning using text propositional relational arguments under a “AI and Cognitive Computing Agents System (AICCAS) framework” using MATLAB. See Research 7.4, section 7.7.2 “Deductive reasoning evaluation for neurologic diseases patients using Cognitive Computing.”
- “Linguistic Neuro-Fuzzing Modeling to analyze different biomedical engineering problem as breast cancer tumor.” The main objective is to obtain a “Linguistic Neuro-Fuzzy model” to detect, analyze, and classify a “breast cancer tumor” as “Benign

Cancer,” a nonaggressive tumor that does not affect surrounding tissues; or *“Malignant Cancer*,” an aggressive tumor that invades surrounding tissues. It can be integrated in an *“AI and Cognitive Computing Agents System (AICCAS) model”* using MATLAB. See Research 7.5, section 7.7.3 *“Linguistic Neuro-Fuzzing Modeling to analyze breast cancer tumor.”*

7.7.1 Research 7.3

“Cognitive sentiments analysis for neurologic diseases that affect mood changes using Cognitive Computing.”

Case for research

Create an *“AI model to detect and classify mood change on neurologic diseases, such as for Parkinson’s disease (PD) patients, and other neurologic diseases, by analyzing their cognitive status using their mood descriptions for yesterday and today.”*

General Objective

Obtain an *“AI model”* to classify the *“mood change”* for yesterday and today for *“neurologic diseases, such as for Parkinson’s disease (PD) patients, and other diseases”* by analyzing their cognitive status using their mood descriptions using an *“AI and Cognitive Computing Agents System (AICCAS) model”* applying MATLAB.

Specific Objectives

- Use *“Word Embedding”* as the collective name for a set of language modeling and feature learning techniques in *“natural language processing,”* where words or phrases from the vocabulary are mapped to vectors of real numbers. to detect positive and negative words.
- Prepare *“Word Embedding”* and partition datafile for classification.
- Train sentiment based on positive/negative words with *“Support Vector Machine.”*
- Test *“sentiment Positive and Negative words”* under *“SVM classification.”*
- Verify word embedding classification accuracy for the *“AI-SVM Model.”*
- Evaluate for PD Patients *“Sentiment Analysis for Yesterday and Today mood”* by text expressions or spoken answers.

Background for *“Parkinson’s mood changes symptom”*

“Parkinson’s” is a *“neurologic disease”* that often starts with a tremor in one hand or leg. Other common *“PD motor symptoms”* are slow movement, stiffness, and loss of balance. But the *“PD nonmotor neurological or cognitive symptoms”* are amnesia, confusion, difficulty in thinking and understanding, mood change, and many more.

Everyone experiences changes in mood over the course of any given day, week, month, and year. But

“mood changes” in Parkinson’s diseases patients are broad term that can mean different things in this disease [37]. The *“mood changes”* usually happen in *“Parkinson’s disease (PD)”* and it is especially important to follow up this cognitive symptom. *“Mood”* refers to a temporary state of mind or generalized state of feeling. In people with *“PD,”* mood can become disordered, with changes that are extreme and persistent or inappropriate to the social context. For this reason *“Parkinson’s”* is also called a *“neuropsychiatric disease.”* This means it is a disease of the *“nervous system (“neuro”)*” that may involve changes in *“mental health (“psychiatric”).”* Emotional and behavioral changes are common in people with chronic diseases, but these changes are even more common in PD. The same *“neurotransmitters, such as the dopamine,”* that regulate movement also regulate our mood. Therefore, the same processes in the brain that lead to the more classical symptoms of Parkinson’s disease can cause depression. When *“dopamine-producing cells in the brain die, movement and mood can also be affected.”* In this case, *“depression”* is actually a symptom of PD, not a reaction to the diagnosis. Besides, others factor can be responsible for the *“mood disorders,”* such as *“drugs interactions,”* and other *“external factors”*:

- *“Drugs interactions.”* Interactions between *“Parkinson’s medications and other drugs”* can also affect mood. It is always important to tell doctors about all the medications and supplements one is taking. This may help to reduce the impact of drug interactions.
- *“External factors,”* such as social, economic, and other factors, in their everyday environment affect their mood. This includes big things like work, family, financial issues, living circumstances, and health, as well as small things like traffic or the weather. *“Stress and anxiety”* are also major triggers for emotional changes [38]

The follow-up for *“PD mood-disorder”* is particularly important because it can be greatly improved with medical treatment to keep patients living their best possible quality of life.

Dataset

A dataset of two lists of positive and negative opinions or sentiment words are used; both lists are known as the *“Opinion Lexicon”* [39]. The lists are: *“positive-words.txt”* and *“negative-words.txt”* containing one field of words, as shown Fig. 7.8. Both lists are used as *“Word Embedding”* to apply *“learning techniques in natural language processing (NLP)”* to evaluate the *“PD patient’s cognitive mood changes”* expressed in two phases: *“one for yesterday and the other for today.”*

Note: This data set can be downloaded in: <http://www.cs.uic.edu> [40], and also can be found in the companion

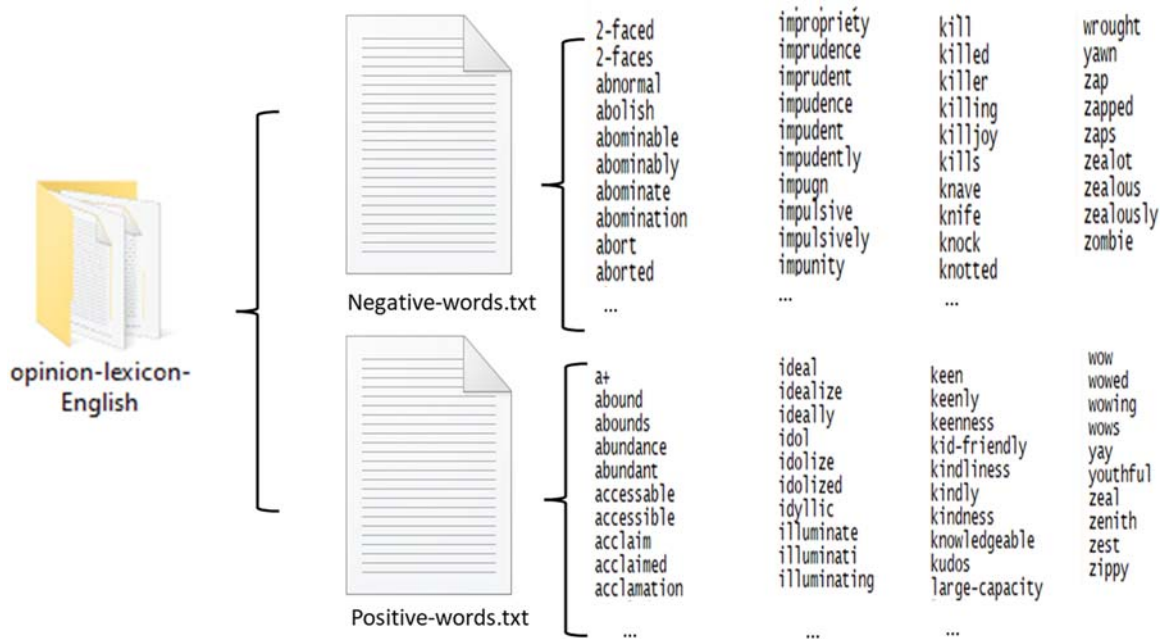


FIGURE 7.8 Dataset sample “Opinion Lexicon” with two text files: negative-words.txt and positive-words.txt.

directory of the book, at the following directory: “. . . \Exercises_book_ABMECH\MATLAB_COGNITIVE \opinion-lexicon-English

Procedure

The steps to “detect and classify mood change on Parkinson’s disease (PD) patients by analyzing their cognitive status via expressing their mood for yesterday and

today using MATLAB*” are summarized in Table of slides 7.3. Each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

Note*: This MATLAB research example requires: Text Analytic Toolbox and Text Analytics Toolbox Model for fast Text English 16 Billion Token Word Embedding.

Table of slides 7.3 Steps for MATLAB to *detect and classify mood change on Parkinson’s disease (PD) patients by analyzing their cognitive status via expressing their mood for yesterday and today.*

Slide	Description	Screen figure
1	<p>Open your MATLAB to “<i>detect and classify PD mood change</i>”—<i>Step (1) Download Word Embedding and “+” and “-” words.</i></p> <p>Go to the directory “... <i>Exercises_book_ABME\CH7\MATLAB_COGNITIVE.</i>” Open the script “<i>smvClassifySentiment.m.</i>” Copy and paste in the command prompt: “<i>step (1) Download WordEmbedding and Positive and Negative words,</i>” that load “<i>fastTextWordEmbedding.</i>” And create a table with the positive and negative words with their respective label.</p>	<p>1. Open your MATLAB to “detect and classify PD mood change” Step 1) Download Word Embedding and “+” & “-” words</p> <p>Go to the directory “... <i>Exercises_book_ABME\CH7\MATLAB_COGNITIVE.</i>” Open the script “<i>smvClassifySentiment.m.</i>” Copy and paste in the command prompt: “<i>step (1) Download WordEmbedding and Positive and Negative words,</i>” that load “<i>fastTextWordEmbedding.</i>” And create a table with the positive and negative words with their respective label.</p> <p>From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

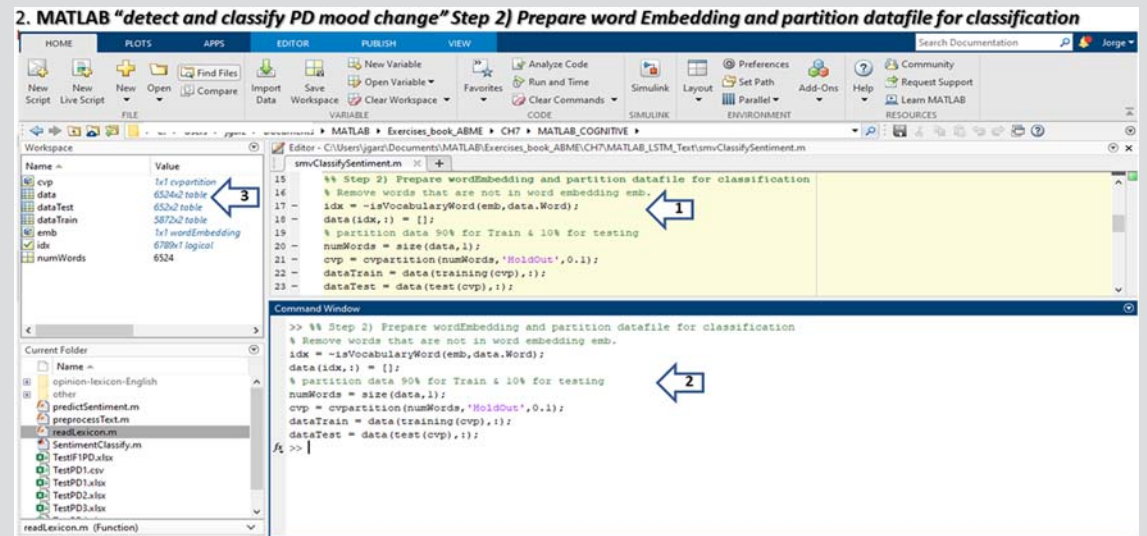
(Continued)

(Continued)

Slide Description

2 MATLAB “detect and classify PD mood change”—Step (2) Prepare word Embedding and partition datafile for classification.
Copy and paste in the command prompt “step (2) Prepare word Embedding and partition datafile for classification.” That remove words that are not found in word embedding variable “emb,” and partition it for “Training Data” and “Testing Data.”

Screen figure



Copy and paste in the command prompt “step (2) Prepare word Embedding and partition datafile for classification”. That remove words that are not found in word embedding variable “emb,” and partition it for “Training Data” & “Testing Data”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 3 MATLAB “detect and classify PD mood change”—Step (3) Train sentiment based positive/negative with Support Vector Machine. Copy and paste in the command prompt “step 3) Train sentiment based positive/negative with Support Vector Machine” and observe the “SVM Model” definition.

3. MATLAB “detect and classify PD mood change” Step 3) Train sentiment based positive/negative with Support Vector Machine

The screenshot displays the MATLAB environment. The Editor window shows the following code in `svmClassifySentiment.m`:

```

25 %% Step 3) Train sentiment based positive/negative with Support Vector Machine
26 % Convert training data convert to word vectors using word2vec.
27 - wordsTrain = dataTrain.Words;
28 - XTrain = word2vec(emb, wordsTrain);
29 - YTrain = dataTrain.Label;
30 % Classify with SVM
31 - mdl = fitcsvm(XTrain, YTrain);

```

The Command Window displays the properties of the trained SVM model:

```

ClassificationSVM
    ResponseName: 'Y'
    CategoricalPredictors: {}
    ClassNames: [Positive Negative]
    ScoreTransform: 'none'
    NumObservations: 5872
    Alpha: [756e+1 single]
    Bias: 0.4030
    KernelParameters: [1e+1 struct]
    BoxConstraints: [5872e+1 double]
    ConvergenceInfo: [1e+1 struct]
    IsSupportVector: [5872e+1 logical]
    Solver: 'SMO'

```

Red arrows in the original image point to the code line `mdl = fitcsvm(XTrain, YTrain);` and the `ClassificationSVM` header in the Command Window.

Copy and paste in the command prompt “step 3) Train sentiment based positive/negative with Support Vector Machine” and observe the “SVM Model” definition

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

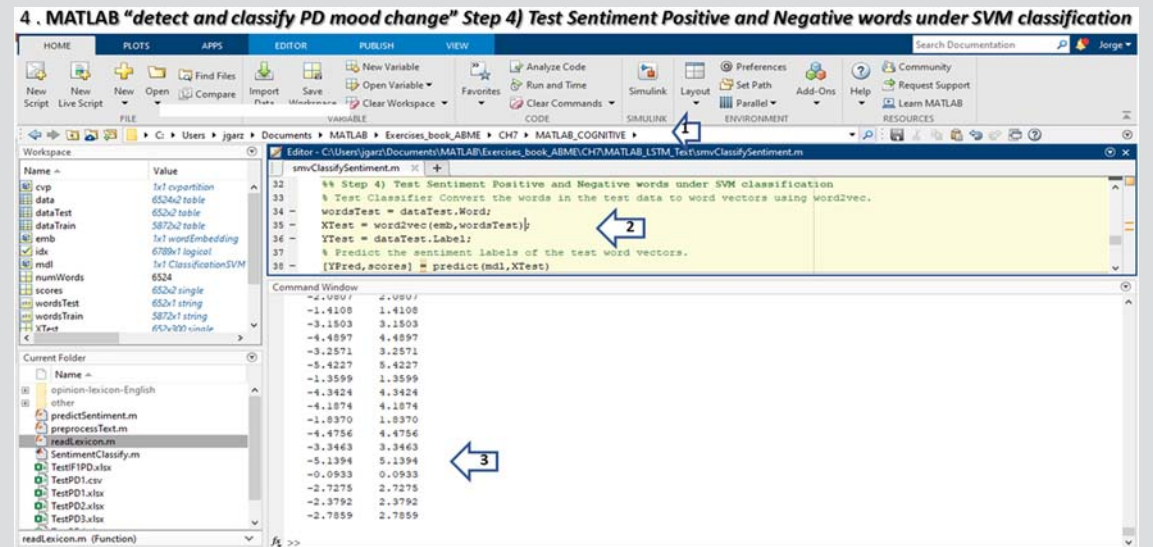
(Continued)

(Continued)

Slide Description

Screen figure

4 MATLAB “detect and classify PD mood change” —Step (4) Test Sentiment Positive and Negative words under SVM classification.
Copy and paste in the command prompt “step (4) Test Sentiment Positive and Negative words under SVM classification,” that convert the words in the testing partition to word vectors, and predict the sentiment labels for the testing partition.



Copy and paste in the command prompt “step (4) Test Sentiment Positive and Negative words under SVM classification”, that convert the words in the testing partition to word vectors, and predict the sentiment labels for the testing partition

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 5 MATLAB “detect and classify PD mood change”—Step (5) Verify word embedding classification accuracy for the AI-SVM Model. Copy and paste in the command prompt “step 5) Verify word embedding classification” accuracy for the AI-SVM Model. That create the confusion chart showing 178 true positive-positive + 444 true negatives-negative with only 11 false negative-positive + 19 false positive-negative that indicate a relation of $622/652 = 0.954$ (95.4% correctly classified).

5. MATLAB “detect and classify PD mood change” Step 5) Verify word embedding classification accuracy for the AI-SVM Model

The screenshot displays the MATLAB environment. The script editor shows the following code:

```

38 [YPred,scores] = predict mdl,XTest;
39 % Step 5) Verify wordembedding classification accuracy for the AI-SVM Model;
40 % Visualize the classification accuracy in a confusion matrix.
41 figure;confusionchart (YTest,YPred);

```

The Command Window shows the execution of the code:

```

>> % Step 5) Verify wordembedding classification accuracy for the AI-
% Visualize the classification accuracy in a confusion matrix.
figure;confusionchart (YTest,YPred);

```

The resulting confusion matrix chart is as follows:

True Class	Predicted Class	
	Positive	Negative
Positive	178	19
Negative	11	444

Copy and paste in the command prompt “step 5) Verify word embedding classification accuracy for the AI-SVM Model”. That create the confusion chart showing 178 true positive-positive + 444 true negatives-negative with only 11 false negative-positive + 19 false positive-negative that indicate a relation of $622/652 = 0.954$ (95.4% correctly classified)

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

6 MATLAB “detect and classify PD mood change”—Step 6) Open file and Calculate Patients Sentiment Analysis Yesterday and Today. Copy and paste in the command prompt “step 6) Open file and Calculate Patients Sentiment Analysis for Yesterday and Today,” select the file: “TestPD1.xlsx.” Observe that original text for yesterday and today “mood,” the number of token (words), the means for each day, and the bagofword for this patient with “mood change from yesterday (negative = -.583) to today (positive) = 0.479”

6. MATLAB “detect and classify PD mood change” Step 6) Open file & Calculate Patients Sentiment Analysis Yesterday and Today

Copy and paste in the command prompt “step 6) Open file & Calculate Patients Sentiment Analysis for Yesterday and Today”, select the file: “TestPD1.xlsx”. Observe that original text for yesterday and today “mood”, the number of token (words), the means for each day, and the bagofword for this patient with “mood change from yesterday(negative=-.583) to today(positive)=0.479”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

7

MATLAB “detect and classify PD mood change”—Step (6) Open file and Calculate Patients Sentiment Analysis Yesterday and Today.

Copy and paste in the command prompt “step (6) Open file and Calculate Patients Sentiment Analysis for Yesterday and Today,” select the file: “TestPD2.xlsx.” Observe that original text for yesterday and today “mood,” the number of token (words), the means for each day, and the bagofword for this patient with “mood change from yesterday (negative = -1.49) to today (positive) = 0.727”

7. MATLAB “detect and classify PD mood change” Step 6) Open file & Calculate Patients Sentiment Analysis Yesterday and Today

The screenshot shows the MATLAB R2020a interface. The Editor window displays the script 'umvClassifySentiment.m' with the following code:

```
42 % Step 6) Open file & Calculate Patients Sentiment Analysis for Yesterday and Today
43 - filename = uigetfile('*.xlsx');
44 - [results,fromTo]=predictSentiment(filename,emb,wordsTrain,mdl);
45 - disp(results);
46 - disp(fromTo);
```

The Command Window shows the output of the script:

```
disp(results);
disp(fromTo);
Original to analyze the Sentiment: Yesterday and Today
*Yesterday, Just diagnosed with very early symptoms.
*Today, I am still working full-time and active. Really need motivation to get more ;

Num. of token or words
2=1 tokenizeDocument:

5 tokens: yesterday just diagnosed early symptoms
14 tokens: today still working full-time active really need motivation get physically

Var1 textData
-1.4908 "Yesterday, Just diagnosed with very early symptoms."
0.7274 "Today, I am still working full-time and active. Really need motivation to get more physically fit to help progr-
```

The output also includes a table of predicted positive and negative sentiments:

Predicted Positive Sentiment	Predicted Negative Sentiment
fulltime active fit help progression today	just diagnosed physically

Copy and paste in the command prompt “step 6) Open file & Calculate Patients Sentiment Analysis for Yesterday and Today,” select the file: “TestPD2.xlsx”. Observe that original text for yesterday and today “mood”, the number of token (words), the means for each day, and the bagofword for this patient with “mood change from yesterday(negative=-1.49) to today(positive)=0.727”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

8 MATLAB “detect and classify PD mood change”—Step (6) Open file and Calculate Patients Sentiment Analysis Yesterday and Today. Copy and paste in the command prompt “step (6) Open file and Calculate Patients Sentiment Analysis for Yesterday and Today,” select the file: “TestPD3.xlsx.” Observe that original text for yesterday and today “mood,” the number of token (words), the means for each day, and the bagofword for this patient with “mood change from yesterday (negative = -.187) to today (negative) = 0.483”

Screen figure

8. MATLAB “detect and classify PD mood change” Step 6) Open file & Calculate Patients Sentiment Analysis Yesterday and Today

The screenshot shows the MATLAB environment with the following components:

- Workspace:** Lists variables such as 'csp', 'data', 'dataTrain', 'emb', 'filename', 'fromTo', 'idx', 'mdl', 'numWords', 'results', and 'scores'.
- Editor:** Contains the MATLAB script 'smvClassifySentiment.m' with the following code:

```
42 % Step 6) Open file & Calculate Patients Sentiment Analysis for Yesterday and Today
43 filename = uigetfile('*.xlsx');
44 [results,fromTo]=predictSentiment(filename,emb,wordsTrain,mdl);
45 disp(results);
46 disp(fromTo);
```
- Command Window:** Shows the execution results:

```
disp(results);
disp(fromTo);
Original to analyze the Sentiment: Yesterday and Today
"Yesterday, Went for acupuncture treatment to deal with lower back pain..."
"Today, it is now worse than when I went to begin with..."
Num. of token or words
2=1 tokenizeDocument:
8 tokens: yesterday went acupuncture treatment deal lower back pain
4 tokens: today worse went begin
Var1
-0.1869 "Yesterday, Went for acupuncture treatment to deal with lower back pain
-0.48323 "Today, it is now worse than when I went to begin with."
```
- File Explorer:** Shows the selection of 'TestPD3.xlsx' from a folder named 'other'.
- Output:** A table of sentiment results is displayed:

Predicted Positive Sentiment	Predicted Negative Sentiment
back treatment begin	yesterday
today	lower acupuncture

Copy and paste in the command prompt “step 6) Open file & Calculate Patients Sentiment Analysis for Yesterday and Today”, select the file: “TestPD3.xlsx”. Observe that original text for yesterday and today “mood”, the number of token (words), the means for each day, and the bagofword for this patient with “mood change from yesterday(negative=-.187) to today(negative)=0.483”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

9

MATLAB “detect and classify PD mood change”—Step (6) Open file and Calculate Patients Sentiment Analysis Yesterday and Today.

Copy and paste in the command prompt “step (6) Open file and Calculate Patients Sentiment Analysis for Yesterday and Today,” select the file: “TestPD4.xlsx.” Observe that original text for yesterday and today “mood,” the number of token (words), the means for each day, and the bagofword for this patient with “mood change from yesterday (negative = -.44) to today (negative) = -0.76”

9. MATLAB “detect and classify PD mood change” Step 6) Open file & Calculate Patients Sentiment Analysis Yesterday and Today

The screenshot displays the MATLAB R2020a environment. The Editor window shows the script `smvClassifySentiment.m` with the following code:

```
42 % Step 6) Open file & Calculate Patients Sentiment Analysis for Yesterday and Today
43 filename = uigetfile('*.xlsx');
44 [results,fromTo]=predictSentiment(filename,emb,wordsTrain,mdl);
45 disp(results);
46 disp(fromTo);
```

The Command Window shows the following output:

```
Original to Analyze the Sentiment: Yesterday and Today
"I Have been flat on my back for 2 days now."
"Today, I am upset to say the least..."

Num. of token or words
2*1 tokenizedDocument:

4 tokens: flat back 2 days
4 tokens: today upset say least

Var1 textData

-0.44026 "I Have been flat on my back for 2 days now."
-0.7555 "Today, I am upset to say the least..."

For TestPD4.xlsx the Sentiment Yesterday(negative) to Today(negative)
```

The File Explorer window shows the selection of `TestPD4.xlsx` in the `Documents` folder.

The word cloud on the right shows the words `today` and `flat days say`.

Copy and paste in the command prompt “step 6) Open file & Calculate Patients Sentiment Analysis for Yesterday and Today,” select the file: “TestPD4.xlsx”. Observe that original text for yesterday and today “mood”, the number of token (words), the means for each day, and the bagofword for this patient with “mood change from yesterday(negative=-.44) to today(negative)=-0.76”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

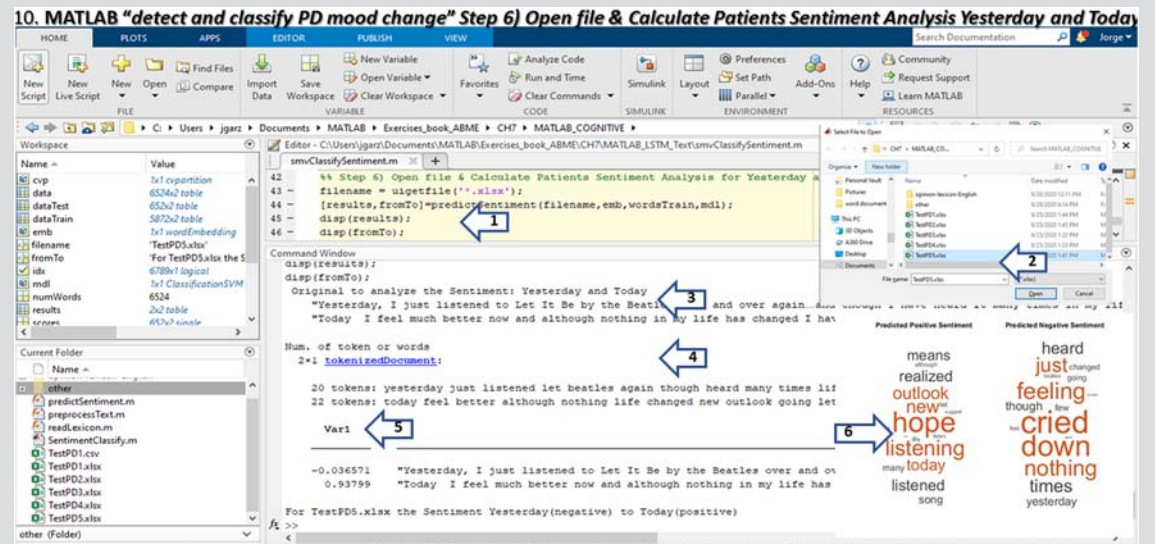
(Continued)

(Continued)

Slide Description

Screen figure

10 MATLAB “detect and classify PD mood change”—Step 6) Open file and Calculate Patients Sentiment Analysis Yesterday and Today.
Copy and paste in the command prompt “step 6) Open file and Calculate Patients Sentiment Analysis for Yesterday and Today,” select the file: “TestPD5.xlsx.” Observe that original text for yesterday and today “mood,” the number of token (words), the means for each day, and the “bag of word” for this patient with “mood change from yesterday (negative = -.037) to today (negative) = 0.937”



Copy and paste in the command prompt “step 6) Open file & Calculate Patients Sentiment Analysis for Yesterday and Today”, select the file: “TestPD5.xlsx”. Observe that original text for yesterday and today “mood”, the number of token (words), the means for each day, and the bagofword for this patient with “mood change from yesterday(negative=-.037) to today(negative)=0.937”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

An “AI model to detect and classify mood change on Parkinson’s disease (PD) patients by analyzing their cognitive status via expressing their mood for yesterday and today” using MATLAB was obtained and it was tested with real answer from “PD patients.” It is possible to create a website or keep a diary where the patients or their care partner can enter the “mood” daily, to obtain a chart to analyze their “mood change” through time as a feedback for whether their medication is improving their quality of life or not; also it can help to detect “stress” and “anxiety.”

Recommendation

It is possible to create a specific “Neurologic opinion with positive and negative frequently used words to describe patient’s behavior with the symptoms labeled.”

7.7.2 Research 7.4

“Deductive reasoning evaluation for neurologic diseases patients using cognitive computing.”

Case for research

Obtain an “CNN-NLP model to detect and classify deductive reasoning evaluation text on neurologic diseases patients, by analyzing their cognitive status based on their answers for three types of deductive reasoning using text propositional relational arguments”.

General Objective

Obtain an “CNN-NLP model” to classify deductive reasoning evaluation on neurologic diseases patients by analyzing their cognitive status based on their answers for three types of “deductive reasoning using text propositional arguments”: “modus ponens,” “modus tollens,” and “disjunction elimination” in the “AI and Cognitive Computing Agents System (A-ICCAS) model” using MATLAB

Specific Objectives

- Use “Word Embedding” as the collective name for a set of language modeling and feature learning techniques in “natural language processing (NLP)” where statements texts using propositional arguments are used to infer the correct answers in all their logical values.
- Three different training dataset are loaded with different examples for “deductive reasoning” for “Modus Ponens Test,” “Modus Tollens Tests,” “Dysfunction Elimination Tests” in tables as a datastores.
- Define a custom “Convolutional Neural Network for NLP classification,” and show its “net graph.”
- Train the network for three types of “Deductive Reasoning using propositional arguments” to obtain three network models, one for each of “deductive reasoning type.”

- Predict new “Deductive reasoning with evaluation tests” for the three different types, showing them in a dialog for user input response of “TRUE” or “FALSE,” and evaluate them using “AI models” to calculate their score percentages results.

Background for “Neurologic—Deductive reasoning evaluation”

As explained in Section 7.6.2 “Deductive reasoning,” and the relationship with “neurologic diseases” in Section 7.6.9, “Cognitive Learning” is a mental process that takes in, interprets, stores, and retrieves information to infer something. It can be studied by the “neuroscience of reasoning” that scientifically involves determining the neural correlation of reasoning that can be investigated by detecting the “event-related potential in the human brain by functional magnetic resonance imaging” [41], with the object of obtaining “AI Models” that apply “Cognitive Computing” methods.

Dataset

The datasets used for this “Deductive reasoning” research are shown in Fig. 7.9. These are:

- Deductive reasoning—Modus ponens: “1testDRModusPonens.csv” and “1trainDRModusPonens.csv”
- Deductive reasoning—Modus tollens: “2testDRModusTollens.csv” and “2trainDRModusTollens.csv”
- Deductive reasoning—Dysfunction elimination: “3testDRDisfunctionElimintion.csv” and “3trainDRDisfunctionElimination.csv”

Note: This dataset can also be found in the companion directory of the book, at the following directory: “. . \Exercises_book_ABME\CH7\MATLAB_DEDUC_REASON”

Procedure

The steps to obtain a “CNN-NLP model” to classify deductive learning evaluation on neurologic diseases patients by analyzing their cognitive status based on their answers for three types of “deductive reasoning using text propositional arguments”: “modus ponens,” “modus tollens,” and “disjunction elimination” in the “AI and Cognitive Computing Agents System (AICCAS) model” using MATLAB* are summarized in Table of slides 7.4. Each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

Note*: This MATLAB research example requires Deep Learning Toolbox, Text Analytic Toolbox, and Text Analytics Toolbox Model for fast Text English 16 Billion Token Word Embedding.

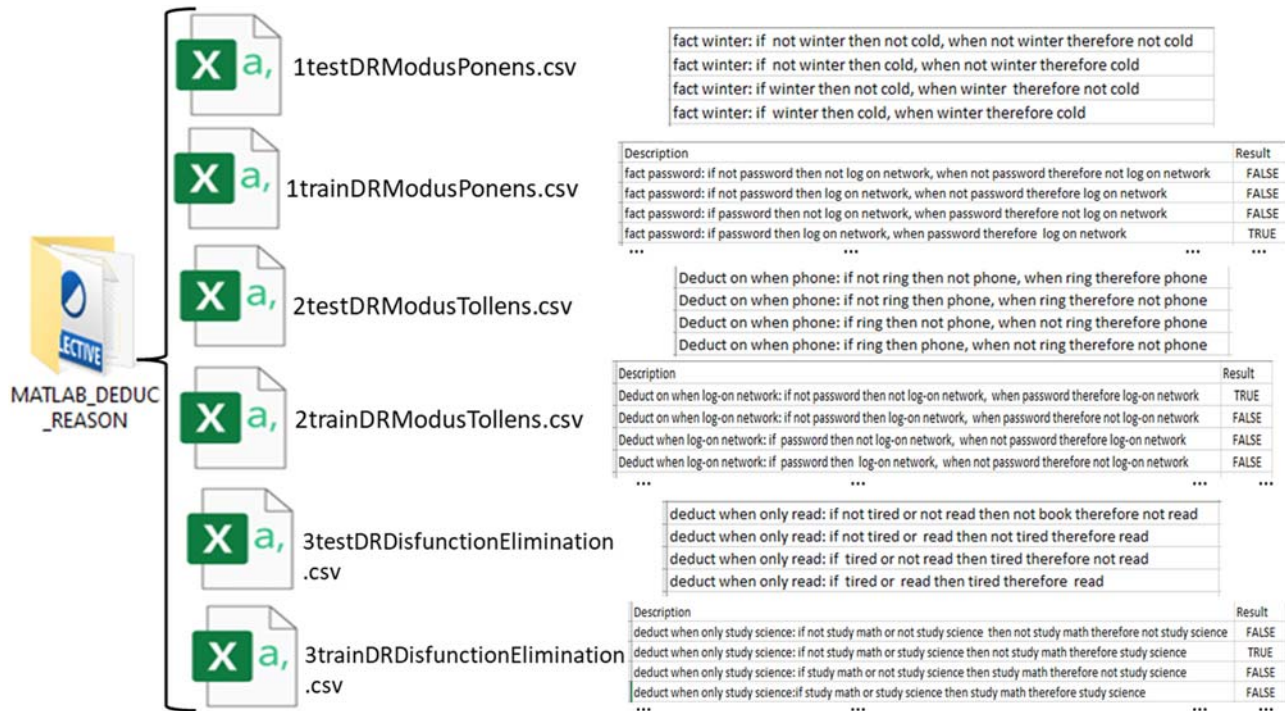
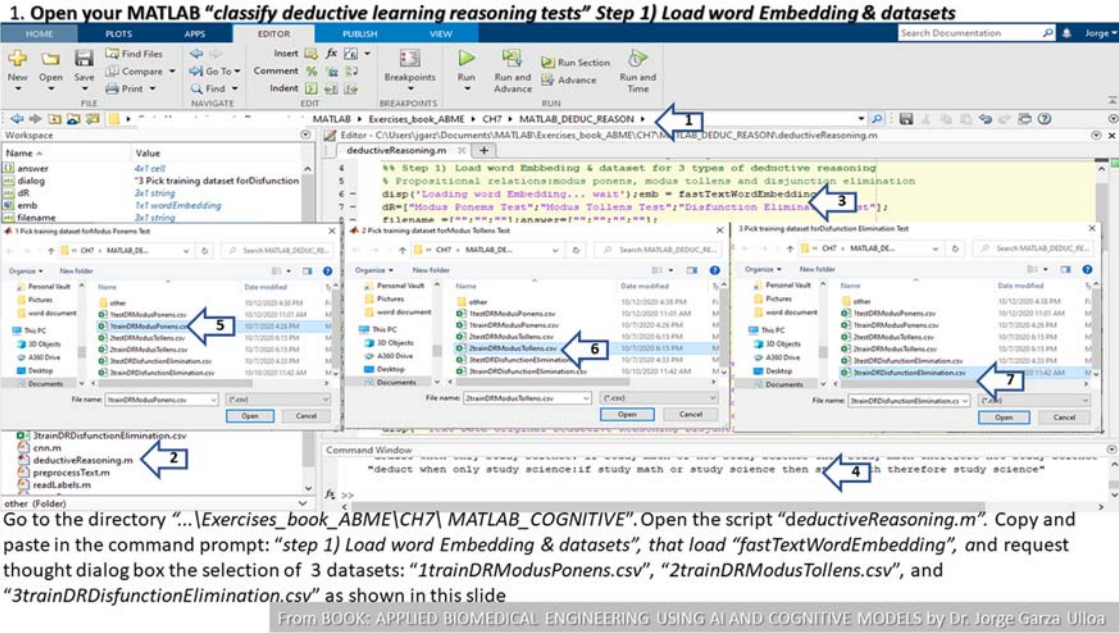


FIGURE 7.9 Datasets for deductive reasoning.

Table of slides 7.4 Steps for MATLAB to *classify deductive learning reasoning tests on neurologic diseases patients analyzing their cognitive status.*

Slide	Description	Screen figure
1	<p>Open your MATLAB “<i>classify deductive learning reasoning tests</i>”—<i>Step 1) Load word Embedding and datasets.</i></p> <p>Go to the directory “<i>... \Exercises_book_ABME\CH7\ MATLAB_COGNITIVE.</i>” Open the script “<i>deductiveReasoning.m.</i>” Copy and paste in the command prompt: “<i>step 1) Load word Embedding and datasets,</i>” that load “<i>fastTextWordEmbedding,</i>” and request thought dialog box the selection of 3 datasets: “<i>1trainDRModusPonens.csv,</i>” “<i>2trainDRModusTollens.csv,</i>” and “<i>3trainDRDisfunctionElimination.csv</i>” as shown in this slide.</p>	 <p>The screenshot shows the MATLAB environment with the script <code>deductiveReasoning.m</code> open. The script contains the following code:</p> <pre> %% Step 1) Load word Embedding & dataset for 3 types of deductive reasoning % Propositional relations:modus ponens, modus tollens and disjunction elimination disp('Loading word Embedding... wait');emb = fastTextWordEmbedding('wordEmbedding.txt'); dS=['Modus Ponens Test';'Modus Tollens Test';'Disfunction Elimination Test']; filename = {'1trainDRModusPonens.csv';'2trainDRModusTollens.csv';'3trainDRDisfunctionElimination.csv'}; </pre> <p>Three file selection dialog boxes are shown, each with a blue arrow pointing to a specific file:</p> <ul style="list-style-type: none"> Dialog 1: Selects <code>1trainDRModusPonens.csv</code> (arrow 5). Dialog 2: Selects <code>2trainDRModusTollens.csv</code> (arrow 6). Dialog 3: Selects <code>3trainDRDisfunctionElimination.csv</code> (arrow 7). <p>The Command Window shows the execution of the script, displaying the output: <code>"deduct when only study science;if study math or study science then h therefore study science"</code> (arrow 4).</p> <p>Below the dialog boxes, a file explorer shows the contents of the directory <code>other</code>, including files like <code>1trainDRDisfunctionElimination.csv</code>, <code>cross.m</code>, <code>deductiveReasoning.m</code>, <code>preprocessText.m</code>, and <code>readLabels.m</code> (arrow 2).</p> <p>At the bottom, the text reads: "Go to the directory “<i>... \Exercises_book_ABME\CH7\ MATLAB_COGNITIVE.</i>” Open the script “<i>deductiveReasoning.m.</i>” Copy and paste in the command prompt: “<i>step 1) Load word Embedding and datasets,</i>” that load “<i>fastTextWordEmbedding,</i>” and request thought dialog box the selection of 3 datasets: “<i>1trainDRModusPonens.csv,</i>” “<i>2trainDRModusTollens.csv,</i>” and “<i>3trainDRDisfunctionElimination.csv</i>” as shown in this slide</p> <p>From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa</p>

(Continued)

(Continued)

Slide Description

Screen figure

2 MATLAB “classify deductive learning reasoning tests”— Step (2) Predictor and response conversion for reasoning learning datasets. Copy and paste in the command prompt “step (3) Predictor and response conversion for the 3 training datastores.” This obtain unique labels, number of observations, transform datastores to text data for “predictors and responses” as shown in this slide.

2. MATLAB “classify deductive learning reasoning tests” Step 2) Predictor & response conversion for reasoning learning

```
25 % Step 2) Predictor and response conversion for the 3 datastores
26 % Obtain unique labels
27 labels1 = readLabels(tdsTrain1,labelName);classNames1 = unique(labels1);
28 labels2 = readLabels(tdsTrain2,labelName);classNames2 = unique(labels1);
29 labels3 = readLabels(tdsTrain3,labelName);classNames3 = unique(labels1);
30 % Obtain number of observations
31 numObservations = numel(labels1); % all data set have the same number of observations
32 % Transform the datastore using transformTextData func. sequence length=14
33 sequenceLength = 14;
34 tdsTrain1 = transform(tdsTrain1, @(data) transformTextData(data,sequenceLength,emb,classNames1));
35 tdsTrain2 = transform(tdsTrain2, @(data) transformTextData(data,sequenceLength,emb,classNames1));
36 tdsTrain3 = transform(tdsTrain3, @(data) transformTextData(data,sequenceLength,emb,classNames1));
37 preview(tdsTrain1) % Preview @ only rows one transformed datastore
```

Predictors	Responses
(1×14×300 single)	FALSE
(1×14×300 single)	FALSE
(1×14×300 single)	FALSE
(1×14×300 single)	TRUE
(1×14×300 single)	FALSE
(1×14×300 single)	FALSE
(1×14×300 single)	FALSE
(1×14×300 single)	TRUE

Copy and paste in the command prompt “step 3) Predictor and response conversion for the 3 training datastores”. This obtain unique labels, number of observations, transform datastores to text data for “predictors and responses” as shown in this slide

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza UJloa

3 MATLAB “classify deductive learning reasoning tests” Step (3) Define a Convolutional Neural Network NLP Reasoning Learning.

Copy and paste in the command prompt “step (3) Define a Convolutional Neural Network NLP” and observe the “CNN graph architecture” defined at the function “cnn.m,” that allows to analyzed two types of responses or classes: “TRUE” and “FALSE.”

3. MATLAB “classify deductive learning reasoning tests” Step 3) Define a Convolutional Neural Network NLP Reasoning Learning

The screenshot displays the MATLAB environment with the following components:

- Workspace:** Lists variables such as `emb` (1x1 wordEmbedding), `filename` (2x1 string), `i` (3), `labelName` ("Result"), `labels1` (48x1 cell), `labels2` (48x1 cell), `labels3` (48x1 cell), `netGraph` (3x1 LayerGraph), `numObservations` (48), `sequenceLength` (14), `tbl1` (48x2 table), and `tbl2` (48x2 table).
- Editor:** Shows MATLAB code for `deductiveReasoning.m`. A yellow highlight covers the following lines:


```
39 %% Step 3) Define a Convolutional Neural Network NLP
40 % create a CNN model for each class of deductive reasoning
41 - netGraph=cnn(emb,sequenceLength,className)
42 - figure
43 - plot(netGraph); % show defined CNN for
44 - title("NLP Convolutional Neural Network")
```
- Command Window:** Shows the execution of the code:


```
>> %% Step 3) Build a Convolutional Neural Net
% create a CNN model for each class of deduct
netGraph=cnn(emb,sequenceLength,className1);
figure
plot(netGraph)
title("NLP Convolutional Neural Network");
% create a CNN model for each class of deduct
netGraph=cnn(emb,sequenceLength,className2);
figure
plot(netGraph); % show defined CNN for two re
title("NLP Convolutional Neural Network");
f3 >>
```
- Graph Visualization:** A graph titled "NLP Convolutional Neural Network" is shown. It features an input layer with nodes `inp1` through `inp3`. These connect to three hidden layers: `con1` (nodes `con1`, `con2`, `con3`), `con2` (nodes `con2`, `con3`, `con4`), and `con3` (nodes `con3`, `con4`, `con5`). The final hidden layer connects to an output layer with nodes `out1`, `out2`, and `classification`. Arrows indicate the flow of information from input to hidden layers and finally to the output.

Copy and paste in the command prompt “step (3) Define a Convolutional Neural Network NLP” and observe the “CNN graph architecture” defined at the function “cnn.m,” that allows to analyzed two types of responses or classes: “TRUE” and “FALSE”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

4 MATLAB “classify deductive learning reasoning tests” Step (4) Train the networks for 3 types of Deductive Reasoning. Copy and paste in the command prompt “step (4) Train the networks for 3 types of Deductive Reasoning,” that train the 3 “Deductive Reasoning tests,” and saved the “AI Model” as “net1 for Modus Ponens Test”, “net2 for Modus Tollens Test” and “net3 for Dysfunction Elimination Test” as shown.

4 . MATLAB “classify deductive learning reasoning tests” Step 4) Train the networks for 3 types of Deductive Reasoning

The figure displays the MATLAB environment. The top window shows the training progress for three networks: net1, net2, and net3. Each network has a 'Training Progress' plot showing performance over iterations (0 to 100). Arrows labeled 4, 5, 6, and 7 point to specific elements in the workspace and plots. The bottom window shows the command prompt with the following code:

```
!Train the network using the trainNetwork function.  
net1 = trainNetwork(sdsTrain1,netGraph,options);  
net2 = trainNetwork(sdsTrain2,netGraph,options);  
net3 = trainNetwork(sdsTrain3,netGraph,options);
```

Copy and paste in the command prompt “step 4) Train the networks for 3 types of Deductive Reasoning”, that train the 3 “Deductive Reasoning tests” , and saved the “AI Model” as “net1 for Modus Ponens Test”, “net2 for Modus Tollens Test” and “net3 for Dysfunction Elimination Test” as shown.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

- 5 MATLAB “classify deductive learning reasoning tests”—
 Step (5) Predict New Deductive reasoning for 3 different types.
 Copy and paste in the command prompt “step (4) Train the networks for 3 types of Deductive Reasoning,” that train the 3 “Deductive Reasoning tests,” and saved the “AI Model” as “net for Modus Ponens Test”, “net2 for Modus Tollens Test” and “net3 for Dysfunction Elimination Test” as shown in this slide, and answer the questions as shown in the next slide.

5. MATLAB “classify deductive learning reasoning tests” Step 5) Predict New Deductive reasoning for 3 different types

The screenshot displays the MATLAB environment with the following components:

- Workspace:** Lists variables such as `predictfilename1`, `predictfilename2`, `predictfilename3`, `predictResponseClean1`, `predictResponseClean2`, `predictResponseClean3`, `scoreMsg1`, `scoreMsg2`, `scoreMsg3`, `scorePoints1`, `scorePoints2`, and `scorePoints3`.
- Editor:** Contains the MATLAB script `deductiveReasoning.m`. Key lines include:


```

      %% Step 5) Predict New Deductive reasoning for 3 different types
      predictfilename1 = uigetfile('*.csv','1 Pick test dataset for Modus Ponens');
      predictfilename2 = uigetfile('*.csv','2 Pick test dataset for Modus Tollens');
      predictfilename3 = uigetfile('*.csv','3 Pick test dataset for Dysfunction Elimination');
      test1=importdata(predictfilename1); % Import tests
      test2=importdata(predictfilename2);
      test3=importdata(predictfilename3);
      predictResponseClean1 = preprocessText(test1,sequenceLength,emb); % Clean Text
      predictResponseClean2 = preprocessText(test2,sequenceLength,emb);
      predictResponseClean3 = preprocessText(test3,sequenceLength,emb);
      % Predict answers based on the models
      [scorePoints1,scoreMsg1]= scoreResponse(test1,dR(1),predictResponseClean1,net1);
      [scorePoints2,scoreMsg2]= scoreResponse(test2,dR(2),predictResponseClean2,net2);
      [scorePoints3,scoreMsg3]= scoreResponse(test3,dR(3),predictResponseClean3,net3);
      final = [scorePoints1,scoreMsg1,scorePoints2,scoreMsg2,scorePoints3,scoreMsg3];
      disp(final);
      
```
- File Selection Windows:** Three windows are open, each with a blue arrow and a number:
 - 1:** Points to the first `uigetfile` call and the first file selection window.
 - 2:** Points to the second `uigetfile` call and the second file selection window.
 - 3:** Points to the third `uigetfile` call and the third file selection window.

Copy and paste in the command prompt “step 5) Predict New Deductive reasoning for 3 different types”. Select the testfiles for each deductive reasoning type: “1testDRModusPonens.csv”, “2testDRModusTollens.csv”, and “3testDRDysfunctionElimination.csv” as shown in this slide, and answer the questions as shown in the next slide

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

6 MATLAB “classify deductive learning reasoning tests” Test Dialogs and results for 3 Deductive reasoning. Answer question for each test as “TRUE” OR “FALSE” and press “OK” button to see the evaluation results as shown. Finally the score for Deductive reasoning for the 3 evaluation tests you are shown in MATLAB command prompt.

6. MATLAB “classify deductive learning reasoning tests” Test Dialogs and results for 3 Deductive Reasoning

Modus Ponens Test

“Deduct when winter: if not winter then not cold, when not winter therefore not cold”
TRUE

“Deduct when winter: if not winter then cold, when not winter therefore cold”
FALSE

“Deduct when winter: if winter then not cold, when winter therefore not cold”
TRUE

“Deduct when winter: if winter then cold, when winter therefore cold”
FALSE

Modus Tollens Test

“Deduct on when phone: if not ring then not phone, when ring therefore phone”
TRUE

“Deduct on when phone: if not ring then phone, when ring therefore not phone”
TRUE

“Deduct on when phone: if ring then not phone, when not ring therefore phone”
FALSE

“Deduct on when phone: if ring then phone, when not ring therefore not phone”
FALSE

Disjunction Elimination Test

deduct when only read: if not tired or not read then not book therefore not read
FALSE

deduct when only read: if not tired or read then not tired therefore read
TRUE

deduct when only read: if tired or not read then tired therefore not read
FALSE

deduct when only read: if tired or read then tired therefore read
TRUE

Command Window

```
Modus Ponens Test
(**Deduct winter: if not winter then not cold, when not winter therefore not cold**)
(**Deduct winter: if not winter then cold, when not winter therefore cold**)
(**Deduct winter: if winter then not cold, when winter therefore not cold**)
(**Deduct winter: if winter then cold, when winter therefore cold**)
Point =>INCORRECT, 1=CORRECT
0
1
0
0
0
*Test: Modus Ponens Test score =100%

Modus Tollens Test
(**Deduct on when phone: if not ring then not phone, when ring therefore phone**)
(**Deduct on when phone: if not ring then phone, when ring therefore not phone**)
(**Deduct on when phone: if ring then not phone, when not ring therefore phone**)
(**Deduct on when phone: if ring then phone, when not ring therefore not phone**)
Point =>INCORRECT, 1=CORRECT
1
0
1
1
1
*Test: Modus Tollens Test score =75%

Disjunction Elimination Test
(**deduct when only read: if not tired or not read then not book therefore not read**)
(**deduct when only read: if not tired or read then not tired therefore read**)
(**deduct when only read: if tired or not read then tired therefore not read**)
(**deduct when only read: if tired or read then tired therefore read**)
Point =>INCORRECT, 1=CORRECT
1
1
1
1
0
*Test: Disjunction Elimination Test score =75%
```

*** Deductive Reasoning evaluation Final score =58.3333%

Answer question for each test as “TRUE” OR “FALSE” and press “OK” button to see the evaluation results as shown. Finally, the Deductive score is shown in this slide.

Conclusions

A “CNN-NLP model” can classify, using deductive learning datasets and predict evaluation testing on neurologic diseases patients, the cognitive status based on answers for three types of “deductive reasoning using text propositional arguments”: “modus ponens,” “modus tollens,” and “disjunction elimination” in the “AI and Cognitive Computing Agents System (AICCAS) model” using MATLAB, as shown in Table of slides 7.4, slide 6.

Recommendation

It is possible to create a connection between “Cognitive Learning” as a mental process that receives, stores, interprets and retrieves information to infer something as a normal process for the human reasoning. This process is studied by “neuroscience of reasoning,” that explain the understading of reasons steps, studied through the neuronal brain activity is detected using imagining bioinstruments as “fMRI”, and others Biomedical instruments.

Note: There are a growing number of studies that show that “COVID-19” is not just a “respiratory disease,” which itself can leave permanent lung scarring with related long-term respiratory issues, but can also affects other human systems, such as it can “attack the central nervous system” as well, leading to neurological issues both during and probably after the virus is controlled. One study has shown that 36% of the autopsies done on COVID-19 victims show that the virus was in the “brain in addition to the lungs and other organs”. [42]. This could be the main reason for the affection of human cognitive identified as a “brain fog in COVID19 long haulers [43].”

7.7.3 Research 7.5

“Linguistic Neuro-Fuzzy Modeling to analyze breast cancer tumor.”

Case for research

Obtain a “Linguistic Neuro-Fuzzy model” to detect, analyze, and classify a “breast cancer tumor” as “Benign Cancer” or “Malignant Cancer.”

General Objective

Obtain an “Linguistic Neuro-Fuzzy model” to detect, analyze, and classify “breast cancer tumor” as “Benign Cancer,” a nonaggressive tumor that does not affect surrounding tissues; or “Malignant Cancer,” an aggressive tumor that invades surrounding tissues. It is to be

integrated in an “AI and Cognitive Computing Agents System (AICCAS) model” using MATLAB

Specific Objectives

- Load the dataset for Breast Cancer for tumor
- Partition dataset into training data and validation
- Create a Mamdani Fuzzy Inference System (FIS) for learning
- Training data and generate FIS rules
- Tune-up optimizing FIS learning updating the rules
- Analyze new data for cancer using the Neuro-Fuzzy Model

Background for “Breast Cancer Tumors”

Read in Chapter 4, section 4.12.2.4.3 the background for “Breast Cancer Tumors” in “research 4.4 Backpropagation Neural Network for Patterns Recognition and classification of Breast Cancer”

Dataset

The dataset used for this research for “Breast Cancer Tumor” is useful for pathologists following the features of biopsies via “Fine Needle Aspiration (FNA)” in accordance with the Wisconsin grade scale that is based on “nine cytological characteristics of breast FNAs.” The values are normalized on a “scale from 0 to 1, with 0 being the closest to benign and 1 the most anaplastic” [44] to determine whether a “breast mass is benign or malign.” The dataset is in one file: “Cancer.csv” with information from 699 instances with nine fields (attributes) as indicated in Table 7.4.

Note: This data is available from the UCI Machine Learning Repository [45].

Procedure

The steps to obtain “Linguistic Neuro-Fuzzy model” to detect, analyze, and classify “breast cancer tumor” as “Benign Cancer” or “Malignant Cancer” that can be integrated in an “AI and Cognitive Computing Agents System (AICCAS) model” using MATLAB are summarized in Table of slides 7.5. Each step of the example is visually explained using screen sequences with instructions in easy to follow figures.

Note*: This MATLAB research example requires: Deep Learning Toolbox. This MATLAB example uses “particle swarm” and “pattern search optimization,” which require the Global Optimization Toolbox software to be installed.

TABLE 7.4 Dataset “Cancer.csv” variable names, fields, and descriptions.

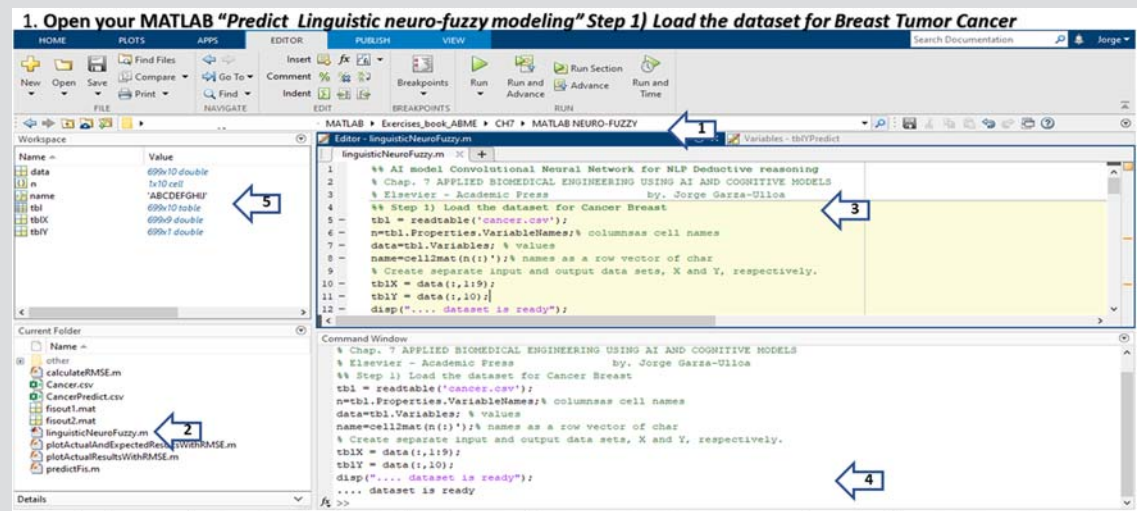
Variable name	Field	Information from 699 instances of cancer tumors, with a Num. of fields or attributes = 9
A	Clump_thickness	Clump thickness (normalized decimal from 0.1 to 1)
B	Uniformity_cell_size	Uniformity of cell size (normalized decimal from 0.1 to 1)
C	Uniformity_cell_shape	Uniformity of cell shape (normalized decimal 0.1 to 1)
D	Marginal_adhesion	Marginal Adhesion (normalized decimal from 0.1 to 1)
E	Single_epithelial_cell_size	Single epithelial cell size (normalized decimal from 0.1 to 1)
F	Bare_nuclei	Bare nuclei (normalized decimal from 0.1 to 1)
G	Bland_chomatin	Bland chomatin (normalized decimal from 0.1 to 1)
H	Normal_nucleoli	Normal nucleoli (normalized decimal from 0.1 to 1)
I	Mitoses	Mitoses (normalized decimal from 0.1 to 1)
j	Malignant_cancer	Malign Cancer ([0 = no, 1 = yes])

Note: This dataset can also be found companion directory of the book, at the following directory: “...!Exercises_book_ABME\CH7\MATLAB NEURO-FUZZY.

Table of slides 7.5 Steps for MATLAB “Linguistic Neuro-Fuzzy model” to detect, classify and predict classification of “breast cancer tumor”

Slide Description Screen figure

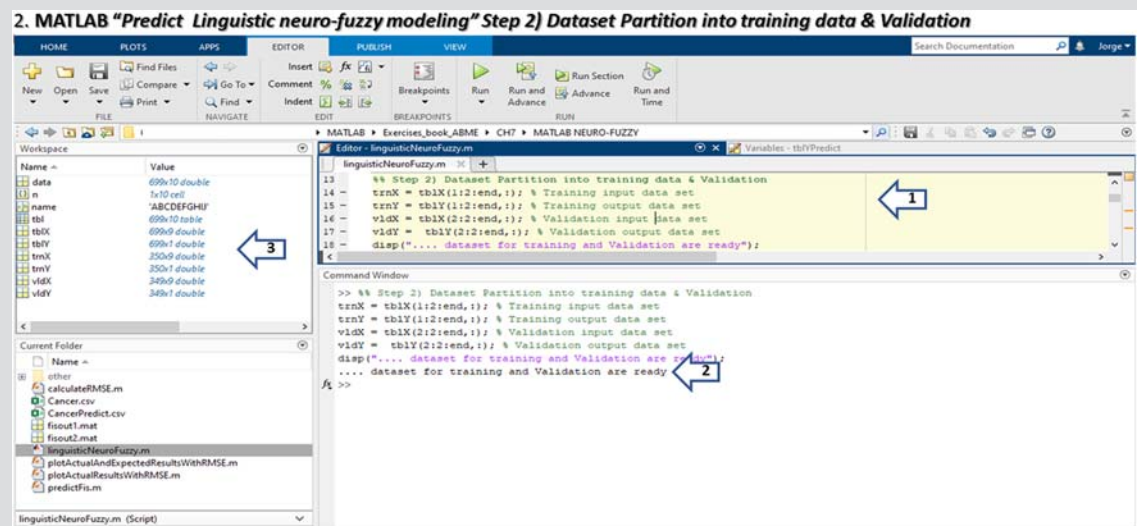
1 Open your MATLAB “Predict Linguistic neuro-fuzzy modeling”—Step (1) Load the dataset for Breast Tumor Cancer.
Go to the directory “... \Exercises_book_ABME\CH7\ MATLAB\MATLAB NEURO-FUZZY. Open the script “linguisticNeuroFuzzy.m.” Copy and paste in the command prompt: “step (1) Load the dataset for Cancer Breast, that load the dataset cancer.csv.” Verify the message: “... dataset is ready.”



Go to the directory “... \Exercises_book_ABME\CH7\ MATLAB\MATLAB NEURO-FUZZY. Open the script “linguisticNeuroFuzzy.m.” Copy and paste in the command prompt: “step 1) Load the dataset for Cancer Breast, that load the dataset cancer.csv”_ Verify the message: “... dataset is ready”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

2 MATLAB “Predict Linguistic neuro-fuzzy modeling”— Step (2) Dataset Partition into training data and Validation.
Copy and paste in the command prompt “step (2) Dataset Partition into training data and Validation.” This step makes the partition in half for training and validation. Verify the message in command prompt: “... dataset for training and Validation are ready.”



Copy and paste in the command prompt “step 2) Dataset Partition into training data & Validation”. This step makes the partition in half for training and validation. Verify the message in command prompt: “... dataset for training and Validation are ready”

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

(Continued)

Slide Description

Screen figure

3 MATLAB “Predict Linguistic neuro-fuzzy modeling”— Step (3) Create a Mamdani FIS for Learning. Copy and paste in the command prompt “step (3) Create a Mamdani FIS for tuning.” This step defines the “Mamdani Fuzzy Inference System (FIS) with 9 inputs each with 2 membership function (MF) and one output with 64 MF”. This allow FIS to use different output MF for each rule. Observe that at this point the FIS structure has zero rules defined.

3. MATLAB “Predict Linguistic neuro-fuzzy modeling” Step 3) Create a Mamdani FIS for Learning

```
%% Step 3) Create a Mamdani FIS for learning.
% Input/output ranges
dataRange = [min(data) 'max(data)'];
fisin = mamfis;
% Add Inputs to FIS
for i = 1:9
    fisin = addInput(fisin,dataRange(i,:), 'Name', name(i));
end
% Add Output to FIS
fisout = addOutput(fisin,dataRange(10,:), 'Name', name(10));
% View the FIS structure. Initially, the FIS has zero rules.
figure
plotfis(fisin)
disp('...Initial FIS structure with zero rules is ready');
```

Copy and paste in the command prompt “step 3) Create a Mamdani FIS for tuning”. This step defines the “Mamdani Fuzzy Inference System (FIS) with 9 inputs each with 2 membership function (MF) and one output with 64 MF”. This allow FIS to use different output MF for each rule. Observe that at this point the FIS structure has zero rules defined.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

4 MATLAB “Predict Linguistic neuro-fuzzy modeling”— Step (4) Training Data and generate FIS rules. Copy and paste in the command prompt “step (4) Training Data and generate FIS rules.” In this step the tune FIS options are specify for “learning” with a “maxIteration = 20” (increase the value for more precision). The results for training, the FIS structure with rules and a sample of the first 5 rules generated are show on the next slide.

4. MATLAB “Predict Linguistic neuro-fuzzy modeling” Step 4) Training Data and generate FIS rules

```
%% Step 4) Training Data and generate FIS rules
options = tuneFisOptions('Method','particle swarm',...
    'OptimizationType','learning', ...
    'NumItakRules',44);
options.MethodOptions.MaxIterations =20;
% Particle swarm optimization uses random search
rng('default')
% Learning rules
fisout1 = tuneFis(fisin,[],trnX,trnY,options);
% View the structure of the tuned FIS, fisout1.
figure
plotfis(fisout1)
disp('... FIS structure with rules is ready. Showing the first five rules');
% Descriptions of the first five rules.
[figout1.Rules(1:5).Description]
% RMSE between generated output data and the validation output data set vty.
limitX=size(vtyX,1);limitY=2;
labelX=' Sample';labelY=' Normalized';
plotActualAndExpectedResultsWithRMSE(fisout1,vtyX,vtyY,limitX,limitY,labelX,labelY);
save fisout1;
```

Copy and paste in the command prompt “step 4) Training Data and generate FIS rules”. In this step the tune FIS options are specify for “learning” with a “maxIteration =20” (increase the value for more precision). The results for training, the FIS structure with rules and a sample of the first 5 rules generated are show on the next slide

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

5 MATLAB “Predict Linguistic neuro-fuzzy modeling”— Step (4) Training Data and check Validation, showing Outputs.

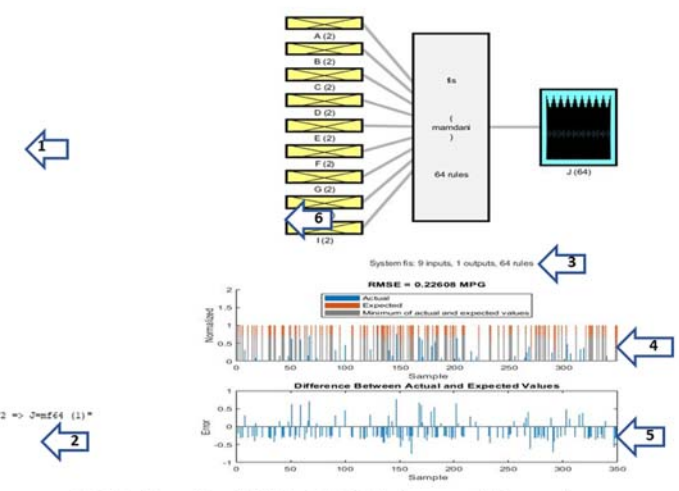
These are the results of the FIS training, the FIS structure with 64 rules with a “RMSE = 0.22608,” the error difference between actual and expect values chart, and a “sample of the first 5 fuzzy rules generated.”

5 . MATLAB “Predict Linguistic neuro-fuzzy modeling” Step 4) Training Data and check Validation – showing Outputs

Iteration	f-count	Best f(x)	Mean f(x)	Stall Iterations
0	100	0.3335	0.5077	0
1	200	0.2793	0.5315	0
2	300	0.2763	0.5077	1
3	400	0.2751	0.5075	0
4	5		8	0
5	600	0.2651	0.4936	0
6	700	0.2651	0.485	1
7	800	0.2491	0.4585	2
8	900	0.2394	0.4541	0
9	1000	0.2523	0.4612	0
10	1100	0.2523	0.4434	1
11	1200	0.2523	0.4699	2
12	1300	0.2394	0.4339	0
13	1400	0.2394	0.4556	1
14	1500	0.2394	0.4523	2
15	1600	0.2394	0.4377	3
16	1700	0.2394	0.4296	4
17	1800	0.2319	0.4122	0
18	1900	0.2319	0.4185	1
19	2000	0.2319	0.4119	2
20	2100	0.228	0.3629	0

Optimization ended: number of iterations exceeded OPTIONS.MaxIterations.
... FIS structure with rules is ready. Showing the first five rules

```
ans =
5x1 string array
*Amf1 & Bmf1 & Cmf2 & Dmf2 & Emf1 & Fmf2 & Gmf2 & Hmf2 => Jmf4 (1)*
*Amf2 & Bmf2 & Dmf2 & Emf2 & Fmf2 & Gmf1 & Hmf1 => Jmf4 (1)*
*Bmf2 & Emf2 & Fmf2 & Gmf1 & Hmf2 => Jmf5 (1)*
*Amf2 & Bmf2 & Gmf2 & Hmf2 => Jmf3 (1)*
*Amf2 & Cmf2 => Jmf2 (1)*
```



These are the results of the FIS training, the FIS structure with 64 rules with a “RMSE=0.22608”, the error difference between actual and expect values chart, and a “sample of the first 5 fuzzy rules generated”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

6 MATLAB “Predict Linguistic neuro-fuzzy modeling”— Step (5) Tune up-Optimizing FIS learning updating the rules.

Copy and paste in the command prompt “step (5) Tune up-Optimizing FIS learning for optimizing the rules.” In this step the FIS option are for: “tuning,” “optimization = patternsearch” and the “MaxInteraction = 30.” Note: these are the recommendable values of interaction greater than used for learning to lower the value of RMSE. This tune process takes longer, and results are shown in the next slide.

6. MATLAB “Predict Linguistic neuro-fuzzy modeling” Step 5) Tune up-Optimizing FIS learning updating the rules

Copy and paste in the command prompt “step (5) Tune up-Optimizing FIS learning for optimizing the rules”. In this step the FIS option are for: “tuning”, “optimization = patternsearch” and the “MaxInteraction=30” . Note: These are the recommendable values of interaction greater than used for learning to lower the value of RMSE . This tune process takes longer, and results are shown in the next slide.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

(Continued)

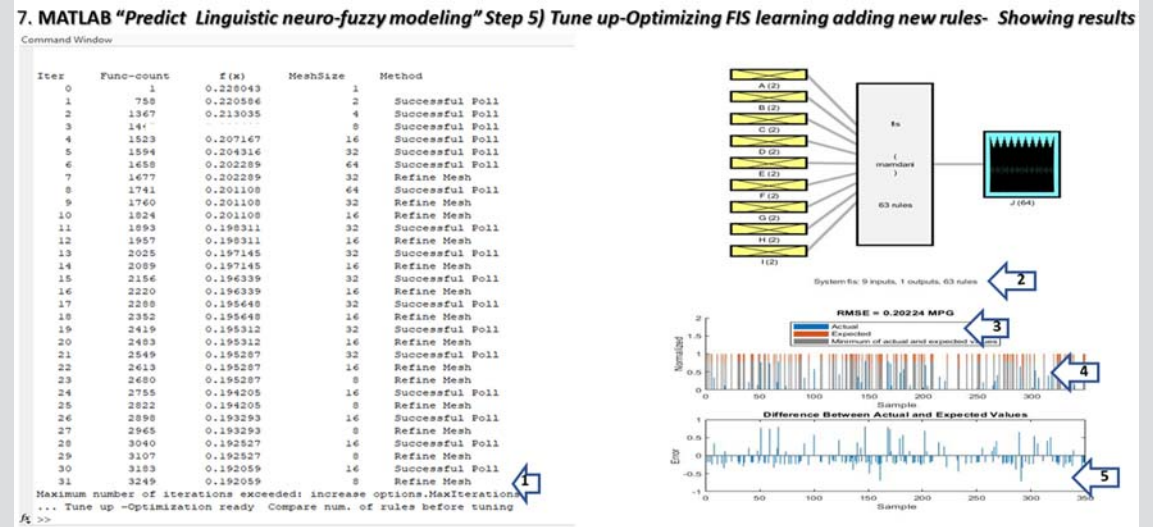
(Continued)

Slide Description

Screen figure

7 MATLAB “Predict Linguistic neuro-fuzzy modeling” -Step (5) Tune up-Optimizing FIS learning adding new rules— Showing results.

Here are the results of the Tune FIS model for optimization: showing the results for the 30 iterations, the FIS structure was optimized for 63 rules for the 64 obtained in learning, the RMSE values is now 0.20224 optimized from RMSE = 0.22608 obtained in learning. And the chart for “Difference between Actual and Expect values using the validation dataset.”

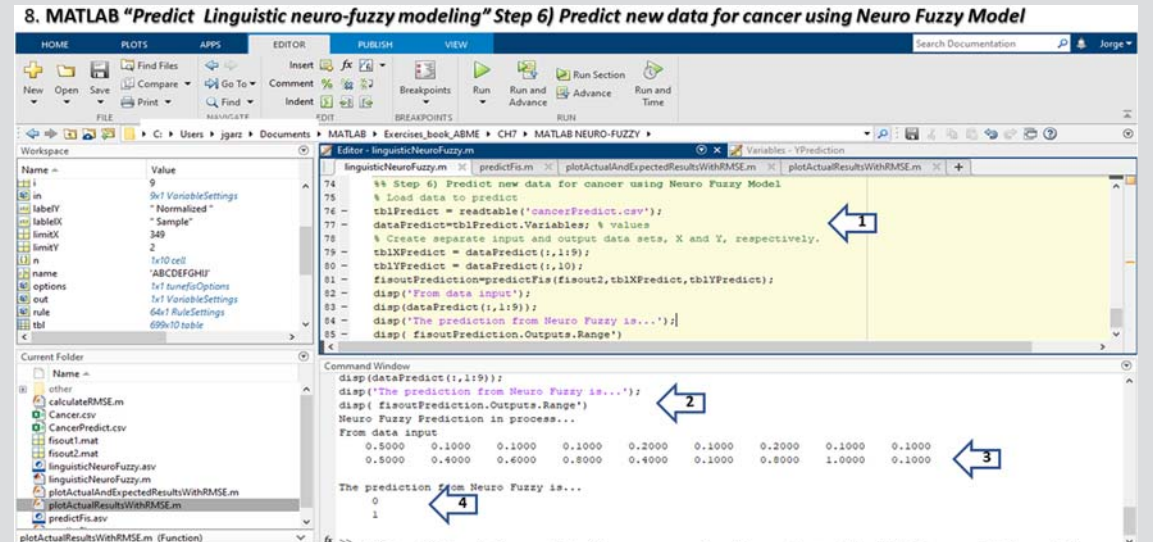


Here are the results of the Tune FIS model for optimization: showing the results for the 30 iterations, the FIS structure was optimized for 63 rules for the 64 obtained in learning, the RMSE values is now 0.20224 optimized from RMSE=0.22608 obtained in learning. And the chart for “Difference between Actual and Expect values using the validation dataset”.

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

8 MATLAB “Predict Linguistic neuro-fuzzy modeling”— Step (6) Predict new data for cancer using Neuro-Fuzzy Model.

Copy and paste in the command prompt “step (6) Predict new data for cancer using Neuro-Fuzzy Model.” The prediction of the new data for “Breast tumor cancer” is based on the optimized FIS model, obtaining the results “0 = no malignant_cancer” for the first patient, and “1 = malignant_cancer” for the second patient



Copy and paste in the command prompt “step (6) Predict new data for cancer using Neuro Fuzzy Model”. The prediction of the new data for “Breast tumor cancer” is based on the optimized FIS model, obtaining the results “0=no malignant_cancer” for the first patient and “1= malignant_cancer” for the second patient

From BOOK: APPLIED BIOMEDICAL ENGINEERING USING AI AND COGNITIVE MODELS by Dr. Jorge Garza Ulloa

Conclusions

A “*Linguistic Neuro-Fuzzy model*” to detect, analyze, classify, and predict the classification of a “*breast cancer tumor*” as “*Benign Cancer*” or “*Malignant Cancer*” was obtained based applying the “*Mamdani-type inference*” as an example that “*AI under fuzzy logic*” can be applied for “*medical diagnostics systems*,” and can be integrated into an “*AI and Cognitive Computing Agents System (AICCAS) framework*”

Recommendation

It is possible to create a connection between “*AI Cognitive Learning*,” and “*neuroscience of reasoning*”. Where, “*Cognitive Learning*” is the mental process that receives, interprets, stores, and retrieves information to infer something during the process of reasoning, and this process can be represented in “*neuroscience of reasoning*,” applying AI as “*Neuro-Fuzzy models*,” as a way to correlates the AI models analysis of the neuronal brain activity obtain through images obtained from many bioinstruments.

7.8 Challenge research for “Applied Biomedical Engineering using Artificial Intelligence and Cognitive Models”

The challenge research are projects with a medium to high degree of difficult. Theses “*challenge projects*” have the objective on cooperate in the development of many solutions to help in different kinds problems applying “*AI and cognitive models*” for different areas of “*Biomedical Engineering**.” All these “*AI models*” proposed once obtained can be integrated in the “*AI and Cognitive Computing Agents System (AICCAS) framework or other AI system architectures shown at Fig. 7.1.*” They will take many hours of research and need of different kind of resources, and must be implemented by an excellent well coordinate multidisciplinary team or individual research that have access to the information needed and dedicated time to resolved them as a “*thesis*” or “*PhD dissertation.*” The best way to realize each of them is defining the specific goals and coordination between all members of the research group involved based in incremental way for every step in their research, design, data collection, intensive data analysis, testing, implementation, debugging, etc.

Sharing information and results with published research are general rules to help world kindness to find more and more useful “*AI models*,” with the main purpose of help people illness, diseases and injuries to find health solutions or at least help to have a better quality of life in patients. I encourage you to register* your project with us at <http://www.garzaulloa.org>, and publish your research, your results, your conclusions, the dataset used, and recommendations to improve these methods or others.

Note*: Feel free to use the proposed projects based on your resources and implement them and/or incorporate new methods to accomplish your objectives and publish them citing this book as one of the references sources.

Once that you choose your challenge project:

Optionally register your research from “*Applied Biomedical Engineering using Artificial Intelligence and Cognitive Models*” sending an email to jorge@garzaulloa.org and jgarzaulloa@miners.utep.edu with the following information:

- *Challenge research project number or your own research,*
- *Project description,*
- *Company, Organization or University doing the research,*
- *State, City and Country,*
- *Name of people working on the project, specialization, and responsibilities with email of each of them,*
- *Design of Experiment summarized,*
- *Timing expected for the realization of the project.*
- *Note: (Optional) Send your datasets if possible.*
- *You will be welcome to ask for help in any question or suggestion about yours projects.*

We will try to provide important tips, recommendations, and other references to achieve your goals.

7.8.1 Challenge research project # 1: “Inductive Reasoning AI evaluation test for neurologic diseases patients under Cognitive Learning and Reasoning applying Cognitive Computing”

Case for research

“*AI-NLP model to detect and classify Inductive Reasoning*” applying AI evaluation tests to healthy people and “*neurologic diseases*” patients, analyzing and compare their “*cognitive status*,” based on their answers to “*inductive reasoning —Bayesian inferences*” using text ad or images as “*statistical arguments.*”

General Objective

Obtain an “*AI-NLP model to detect and classify Inductive Reasoning*” applying evaluation tests to healthy people and neurologic diseases patients, analyzing and compare their “*cognitive status*” based on their answers to “*Inductive Reasoning —Bayesian Inferences*” using “*Statistical Arguments*” of three types: “*Inductive Learning and reasoning with generalized arguments*,” “*Vision-Memory-Language as generalized arguments*,” and “*Inferring causal relations*” to be integrated under the “*AI and Cognitive Computing Agents System (AICCAS) framework*” as shown in Fig. 7.1

Specific Objectives

- Build a dataset for “*Inductive Reasoning —Bayesian Inferences*” using text “*Statistical Arguments*” of three types: “*Inductive Learning and reasoning with generalized arguments*,” “*Vision-Memory-Language as generalized arguments*,” and “*Inferring causal relations*”
- Use a “*Word Embedding*” as the collective name for a set of language modeling and feature learning techniques in “*natural language processing (NLP)*,” where “*if statements texts using propositional arguments*” are used to infer the correct answers in all their logical values.
- Define a three custom “*AI models*,” one for each of the three types needed: “*Inductive Learning and reasoning with generalized arguments*,” “*Vision-Memory-Language as generalized arguments*,” and “*Inferring causal relations*,” and shows their “*net graphs*”
- Train the network for three types of “*Inductive Reasoning using propositional arguments*” to obtain three3 network models variables, one for each of “*inductive reasoning type*.”
- Predict new “*inductive reasoning*” with evaluation tests for the three different types, shown them in a dialog or screen for user input and evaluate them by the “*AI models*” obtained calculating their score percentages results.
- Compare areas and brain activities between health people and neurologic diseases patients

Background for Bayesian models in cognitive science

“*Bayesian models in cognitive science*” can be obtained in different ways as evaluating experience with “*probability and statistical values*” as shown in the following research reference papers [46–48]: “*Inductive Learning and reasoning*,” “*Vision-Memory-Language*,” and “*Inferring causal relations*.” Where:

- “*Inductive Learning and reasoning with generalized arguments*” can be based on relational arguments, i.e.
 - Input arguments: “*Cows can get Kick’ disease, and gorillas can get Hick’s disease*”
 - Conclusion argument: “*All mammals get Hick’s disease*”
 - Task: “*Judge how likely conclusion is to be true, given that arguments are true*”
- “*Vision-Memory-Language as generalized arguments*” allows to learn concepts and words memorizing their relation between them with images from objects, for example:
 - Input arguments: A series of “*images of dog*” to recognize “*dogs*,”
 - Conclusion arguments: Place a group of images from different animals
 - Task: recognize probabilities which are from “*dogs*,” and which are not from “*dogs*.”

- “*Inferring causal relations*” form previous experiences as the following:

- Input arguments:

- Day 1: I took a multivitamins capsule, and I had not a headache
- Day 2: I took a multivitamins capsule, and I had a headache
- Day 3: I did not take a multivitamins capsule, and I had a headache
- Day 4: I took a multivitamins capsule, and I had not a headache
- **Conclusion causal argument: *Multivitamins capsules causes headache***

- Task: “*Judge the probability of a causal link between taking multivitamins capsules and headache*”

Note: Review [Section 7.6.3](#) Inductive Reasoning for more information

In neuroimaging previous studies report

- Paper: “*Prefrontal and parietal activity is modulated by the rule complexity of inductive reasoning and can be predicted by a cognitive model*” [49] A cognitive model predicts both the behavioral and brain imaging results. The neural activity in the right “*right dorsal lateral prefrontal cortex (DLPFC)*,” and “*precuneus*” is modulated by rule complexity. Increased task complexity can lead to increased activation in task-specific regions or to activation of additional regions. How the brain adapts to increased rule complexity during inductive reasoning remains unclear. Increased activations accompany increased rule complexity in the “*DLPFC*” and “*medial posterior parietal cortex (precuneus)*.” A “*cognitive model*” predicted both the behavioral and brain imaging results. The current findings suggest that neural activity in frontal and parietal regions is modulated by rule complexity, which may shed light on the neural mechanisms of inductive reasoning.
- Paper: “*Visualization, inductive reasoning, and memory span as components of fluid intelligence*” [50]: Implications for technology education, support research on learning in technology education, this paper describes two studies which aimed to identify cognitive factors which are components of fluid intelligence. The results identify that a synthesis of visualization, short-term memory span and inductive reasoning can account for approximately 28%– 43% of the variance in fluid intelligence. A theoretical rationale for the importance of these factors in technology education is provided with a discussion for their future consideration in cognitive interventions. As discuss in the research paper: “*Data-Brain driven systematic human brain data analysis: A case study in numerical inductive reasoning centric investigation*”

[51]. As a crucial step in understanding human intelligence, “Brain Informatics (BI)” focuses on thinking centric investigations of human cognitive functions with respect to multiple activated brain areas and neurobiological processes for a given task. In this paper, they propose a “Data-Brain driven approach for systematic brain data analysis,” which is implemented by using the “Data-Brain,” “Data-Brain based BI provenances,” and “Global Learning Scheme for BI.” Furthermore, a human numerical “inductive reasoning” centric investigation is described to demonstrate significance and usefulness of the proposed approach. Such a “Data-Brain driven approach” reduces the dependency on individual capabilities and provides a practical way for realizing the systematic human brain data analysis of BI methodology.

Dataset

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be obtained after the procedures and tests

Recommendation

To be analyzed after results for the assigned research team.

7.8.2 Challenge research project # 2: “Abducting Reasoning using AI evaluation tests for patients under Cognitive Learning and Reasoning applying Cognitive Computing”

Case for research

“AI-NLP models to detect and classify Abducting Reasoning using causal arguments in clinical diagnostic reasoning for patients using AI evaluation tests under Cognitive Computing.”

General Objective

Obtain “AI models to analysis metaphoric reasoning for clinical diagnosis using Cognitive Computing based on 4 approaches of Metaphoric reasoning-causal arguments (Cause \supset Effect): “Inference of abduction” based on patterns “IF/THEN/THEREFORE,” “Backward-chaining” as a Query-Driven using AND-OR search through the space of explanations, “Paradigm case-based” as cases representation to reach explanations, “Generative metric” based on query human inferences.

Specific Objectives

- Create the data structures needed as “(Cause \supset Effect),”
- Build the dataset for each approach: “Inference of abduction,” “Backward-chaining,” “Paradigm case-based,” “Generative metric”

- Define the customs “Convolutional Neural Network for NLP classification” needed for each approach, and shows their “net graph”
- Train the network for four approaches of “Abductive Reasoning using causal arguments” to obtain the four “AI models.”
- Predict new “Abductive reasoning” correct answers for the evaluation test for the four approaches and evaluate them for each patient calculating their score percentages results.

Background

An ideal version of diagnosis using circular “abductive reasoning with causal arguments” as explained in section 7.6.4 is a framework as shown in Fig. 7.4, that includes “Knowledge Storage AI database (KSAID),” where clinical information is stored as related information as “(Cause \supset Effect),” that is, “(disease \supset symptoms),” “(disease \supset causes),” “(disease \supset diagnosis),” “(disease \supset treatments),” etc. as explained in the example in that section. Use “Cognitive Computing” based on the four approaches of “Metaphoric reasoning-causal arguments (Cause \supset Effect)”: “Inference of abduction” based on patterns “IF/THEN/THEREFORE,” “Backward-chaining” as a “Query-Driven” using an “AND-OR” search through the space of explanations, “Paradigm case-based” as case representations to reach explanations, and “Generative metric” based on query human inferences.

Dataset

The datasets for each specific approach can be built for each kind of approaches, and can be used to train and validate the “AI models,” to be used to predict the answers for the different tests needed under cognitive computing methods

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be obtained after the procedures and tests

Recommendation

To be analyzed after results for the assigned research team.

7.8.3 Challenge research project # 3: “Metaphoric reasoning for clinical diagnosis using Cognitive Learning and Reasoning applying Cognitive Computing”

Case for research

Obtain “AI model to analysis metaphoric reasoning-causal arguments under Metaphorical inference reasoning for clinical diagnosis applied to neurologic disease using Cognitive Learning and Reasoning applying Cognitive Computing”

General Objective

Obtain “AI model to analysis metaphoric reasoning for clinical diagnosis for a Parkinson’s diseases and Parkinsonism using Cognitive Computing based on approaches of Metaphorical inference reasoning-causal arguments ($Cause \supset Effect$)” based on: signs and symptoms, medical history reviewed by neurologist, and physical examination.

Specific Objectives

- Concentrate current information needed at “Knowledge Storage AI Database (KSAIDB)” as explained in section 7.2 about *Parkinson’s diseases and Parkinsonism*
- Apply “Metaphorical inference reasoning” using the four-stage process: “Recalling,” “Mapping,” “Testing,” and “Recurrences” as explained at section 7.6.6
- Find “reusable rules” for similarities between “neurologic diseases” as “Parkinson’s diseases and Parkinsonism”
- Obtain an “AI model” from train datasets and validate datasets to analysis “metaphoric reasoning” in clinical diagnosis based on similarities between two neurologic diseases
- Test the “AI Model” obtained with other examples on similarities between two neurologic diseases

Background

“Parkinson’s Disease (PD)” is a progressive neurodegenerative disease producing “neuronal cell death,” presenting “loss of dopamine production in the brain area” known as “substantia nigra,” altering the “central nervous system (CNS),” and affecting the regulation of the “human movements and emotions.” The exact cause of this damage is still unknown, and currently there is no cure for “Parkinson’s disease.” “PD” is a form of extrapyramidal disorders that affects movements disorder caused by damage to the “extrapyramidal tract,” a network of nerves that controls movements [51]. “PD” signs and symptoms can be different for everyone. Symptoms often begin on one side of your body and usually remain worse on that side, even after symptoms begin to affect both sides. These signs and symptoms may include: “tremor,” “slowed movement (bradykinesia),” “rigid muscles,” “impaired posture and balance,” “loss of automatic movements,” “speech changes,” “writing changes,” and others.

“Parkinsonism” is a general term that refers to a group of neurological disorders that cause movement problems similar to those seen in “Parkinson’s disease” such as “tremor,” “slow movement,” “impaired speech,” or “muscle stiffness” especially resulting from the “loss of dopamine-containing nerve cells (neurons)” [52]. There are many other causes of “parkinsonism” including:

“Medications,” “Repeated head trauma,” “Certain neurodegenerative disorders,” “Exposure to toxins,” “Certain brain lesions,” “Metabolic and other disorders,” and others.

Not everyone who has “Parkinsonism” has “Parkinson’s disease.” There is overlap in treatment for “Parkinson’s and parkinsonisms.” “Dopaminergic therapy” is typically the first line treatment for Parkinson’s, and also can be effective in some “parkinsonisms” [53].

It is important to know that many people will not exhibit the cardinal symptoms necessary for a diagnosis of a specific disorder and will simply be labeled as “parkinsonism.” Under the category of “parkinsonism” there are a number of disorders, some of which have yet to be clearly defined or named.

Early in the disease process, it is often hard to know whether a person has “idiopathic (unknown origins) Parkinson’s disease” or a syndrome that mimics it. “Parkinsonism” is also known as “atypical Parkinson’s disease,” represent about 10%–15% of all diagnosed cases of “parkinsonism.” These syndromes “tend to progress more rapidly than Parkinson’s,” present with additional symptoms such as “early falling,” “dementia” or “hallucinations,” and “do not respond or respond only for a short time to levodopa therapy.”

Dataset

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be obtained after the procedures and tests

Recommendation

To be analyzed after results for the assigned research team.

7.8.4 Challenge research project # 4 “Neurologic – evaluating anxiety in neurologic diseases using Cognitive Therapy Theory using Cognitive Learning and Reasoning with Cognitive Computing”

Case for research

“Neurologic – evaluating anxiety in neurologic diseases using Cognitive Therapy Theory applying Cognitive Learning and Reasoning with Cognitive Computing models”

General Objective

Obtain an “AI model” to classify the level of “anxiety” based on generations of thoughts arguments that increase “anxiety” in patients with and without neurologic disease analyzing their answers applying cognitive therapy tests, to obtain an anxiety score, and if it is possible correlate with detection and measurements of brain areas activities with

Biomedical images using “*neuroscience of reasoning*” methods, to obtain a “*Cognitive Computing Model*” to be integrated in the “*AI and Cognitive Computing Agents System (AI-CCAS) framework*.” as shown in Fig. 7.1

Specific Objectives

- Develop new datasets for “*Cognitive Therapy tests*,” with patterns that can be detected by “*AI models*,” as explained in section 7.6.8 or apply the exiting specialized test [54–56], and many more
- Apply “*Natural Language Processing*” methods as: Word Embedding, preprocessing of text etc.
- Define a custom “*Convolutional Neural Network for NLP classification*,” and “*other Deep Learning models to process images of brain activity*”
- Training the “*AI models*”
- Evaluate the answer using the “*AI models*” comparing results with health people and neurologic diseases patients

Note*: Synchronizing the application in real time images with the using “*Cognitive Therapy Theory to test Cognitive Computing Model*”

Background

Cognitive Therapy Theory (CTT)” is based on the “*cognitive model*,” which states that thoughts, feelings and behavior are all connected, and that individuals can move toward overcoming difficulties, and meeting their goals by identifying and changing unhelpful or inaccurate thinking, problematic behavior, and distressing emotional responses. In general term “*CITT*” is based that the root cause of why most people experience “anxiety is due to thoughts and not the situation.” Nevertheless, millions of people with anxiety have physical symptoms as: tingling hands and feet, nerve pains, lightheadedness/dizziness, headaches, vision problems, fatigue, memory loss, confusion, etc., these resemble neurologic diseases such as: “*Multiple Sclerosis*,” “*Brain Tumors*,” “*Lyme Diseases*,” and others. Unfortunately, there is not a simple way to detect the difference between suffering from anxiety or having a neurologic disorder.

Some example of a “*Cognitive behavioral theory anxiety tests*” are:

- Situation: You are stuck in heavy traffic going to the office
- Thought 1: You are going to be late as sometimes happen and your boss is always pressuring to be on time
- Thought 2: “*I have to call my boss, and explain let him know what is happening*”
- Thought 3: “*I have to look for an alternative route to save time*”
- Thought 4: “*I cannot believe it, it is happening again, I’m going to lose my job!*”
- Many other thoughts

Analyze answer as:

- “Is this thought logical?”
- “Is this thought rational?”
- “Are there a way other people might see and think about this same event that is different than how I’m seeing it?”
- Many others alternative to diminish the “*anxiety*.”

Some researchers have focus in determine:

- The effect of cognitive behavioral therapy in the in the management of depression in type 2 DM patients [57].
- Examined fidelity measurement for “*cognitive processing therapy (CPT) for posttraumatic stress disorder (PTSD)*” [58].
- Parkinson’s Disease and Bilateral Subthalamic Nuclei Deep Brain Stimulation: Beneficial Effects of Preoperative Cognitive Restructuration Therapy on Postoperative Social Adjustment [59]
- Cognitive behavioral therapy for depression and anxiety of Parkinson’s disease: A systematic review and meta-analysis [60]
- And many others.

Dataset

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be obtained after the procedures and tests

Recommendation

To be analyzed after results for the assigned research team.

7.8.5 Challenge research project # 5: Analyze Neurologic opinion words with positive and negative frequently used to describe patient’s behavior with the symptoms labeled”

Hint: See introduction to the research project see research project 7.3

Case for research

Obtain “*AI Models*” to “*analyze neurologic opinion words to be classified with positive and negative frequently used to describe patient’s behavior with the symptoms labeled*” applying “*Cognitive Learning and its relationship with neuroscience of reasoning proposed as Cognitive Learning- Reasoning (CL&R) using Cognitive Computing (CC)*” using the “*AI and Cognitive Computing Agents System (AI-CCAS) framework*” as shown in Fig. 7.1

General Objective

Develop the necessary “*AI-Models*” based on the possibility of “*creating specific Neurologic opinion words*” to be classify as positive and negative frequently used to describe patient’s behavior with the symptoms labeled” applying “*Cognitive Learning and its relationship with*

neuroscience of reasoning proposed as Cognitive Learning- Reasoning (CL&R) using Cognitive Computing (CC) using the “*AI and Cognitive Computing Agents System (AI-CCAS) framework.*”

Specific Objectives

To be define for assigned research team.

Background

To be define for assigned research team.

Dataset

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be define for assigned research team.

Recommendation

To be define for assigned research team.

7.8.6 Challenge research project # 6: “Classify status of neurologic disease patients analyzing their images, movements in real time video, and speech

Case for research

Obtain “*AI models to classify status of neurologic patients analyzing their images, movements in real time video, and speech*” applying “*Cognitive Learning and its relationship with neuroscience of reasoning proposed as Cognitive Learning- Reasoning (CL&R) using Cognitive Computing (CC)*” using the “*AI and Cognitive Computing Agents System (AI-CCAS) framework*” as shown in [Fig. 7.1](#)

General Objective

- “*AI model*” for speech using NLP
- “*AI model*” for images using “*ANN*” as explained at Chapter 5 Deep Learning Models Principles Applied to Biomedical Engineering
- “*AI model*” for images using “*ANN*” as explained at Chapter 6 Deep Learning Models Evolution Applied to Biomedical Engineering

Specific Objectives

To be define for assigned research team.

Background

Read background of 7.7.3 Challenge research project # 3: “*Metaphoric reasoning for clinical diagnosis using Cognitive Learning and Reasoning applying Cognitive Computing*”

To be define for assigned research team.

Hints:

1. Read Chapter 7 of my book: “*Applied Biomechanics using mathematical models*” [10]
2. For speech it is possible to analyze pitch and “*Mel-frequency cepstral coefficients (MFCC)*,”
3. For images and video analysis apply “*ANN*” models studied in this book

Dataset

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be define for assigned research team.

Recommendation

To be define for assigned research team.

7.8.7 Challenge research project # 7: “human voice cognitive analysis for cognitive services as voice therapy”

Case for research

Obtain “*AI-Models*” for the human voice cognitive analysis for cognitive services as voice therapy, applying “*Cognitive Learning and its relationship with neuroscience of reasoning proposed as Cognitive Learning- Reasoning (CL&R) using Cognitive Computing (CC)*” using the “*AI and Cognitive Computing Agents System (AI-CCAS) framework*” as shown in [Fig. 7.1](#)

General Objective

Develop “*AI tools*” for “*Voice therapy*,” that consists of techniques and procedures that target vocal parameters, such as vocal fold closure, pitch, volume, and quality. This therapy is provided by “*speech-language pathologists and is primarily used to aid in the management of voice disorders, or for altering the overall quality of voice, as in the case of transgender voice therapy is a related field to alter voice for the purpose of singing.*” “*Voice therapy*” may also serve to teach preventive measures such as vocal hygiene and other safe speaking or singing practices.

Specific Objectives

To be define for assigned research team.

Background

The human voice reflects both physical and psychic states and has the ability to convey both cognitive meaning and affective expression simultaneously. It is our primary mode of communication for both ideas and feelings and can move us with words and beyond words. The purpose and goals of voice therapy are to help in develop an awareness of the “*new*” voice. Discern the “*new*” voice from the “*old*” voice. Integrate this modified voice quality into conversational speech in an automatic and consistent manner, without conscious use of the voice therapy technique.

Dataset

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be define for assigned research team.

Recommendation

To be define for assigned research team.

7.8.8 Challenge research project # 8: Cognitive Behavioral Therapy

Case for research

Develop “AI models” to be used on “Cognitive Behavioral Therapy” applying “Cognitive Learning and its relationship with neuroscience of reasoning proposed as Cognitive Learning- Reasoning (CL&R) using Cognitive Computing (CC)” using the “AI and Cognitive Computing Agents System (AI-CCAS) framework” as shown in Fig. 7.1

General Objective

Develop “AI applications” for “Cognitive behavioral therapy (CBT),” as explain in section 7.8.4 is a psycho-social intervention that aims to improve mental health. “CBT” focuses on challenging and changing unhelpful cognitive distortions and behaviors, improving emotional regulation, and the development of personal coping strategies that target solving current problems. Originally, it was designed to treat depression, but its uses have been expanded to include treatment of many mental health conditions, including anxiety. “CBT” includes several cognitive or behavior psychotherapies that treat defined psychopathologies using evidence-based techniques and strategies

Specific Objectives

To be define for assigned research team.

Background

To be define for assigned research team.

Dataset

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be define for assigned research team.

Recommendation

To be define for assigned research team.

7.8.9 Challenge research project # 9: detection of “prefatigue/fatigue by stress and anxiety”

Case for research

Develop “AI models” to be used on “prefatigue/fatigue” by stress and anxiety” applying “Cognitive Learning and its relationship with neuroscience of reasoning proposed as Cognitive Learning- Reasoning (CL&R) using Cognitive Computing (CC)” using the “AI and Cognitive Computing Agents System (AI-CCAS) framework” as shown in Fig. 7.1

General Objective

For example, on this “Coronavirus Pandemic”, that has affected us on many fronts: health, economy, independence, free movement, education, and many things. In this research we could focus on something that affects

us all and if we do not recognize it, it will continue to envelop and destroy one of the most precious things in our lives our “human mind” through “*prefatigue of COVID-19.*” Then is necessary to develop methods to detect of “*prefatigue/fatigue*” by “*prefatigue of COVID-19.*” applying “Cognitive Learning and its relationship with neuroscience of reasoning proposed as Cognitive Learning- Reasoning (CL&R) using Cognitive Computing (CC)” see section 7.6.9

Specific Objectives

The detection of “prefatigue/fatigue” can be based of “*cognitive types*” symptoms, that can be detected by “Cognitive Learning- Reasoning (CL&R) using Cognitive Computing (CC)” as:

- “Increases response time,”
- “Reduction of attention,”
- “Deteriorated memory,”
- “Withdrawn mood,”
- “Very poor decision-making,”
- “Slow reaction to changing situations,”
- “Failures of responses to an imminent conflict or danger,”
- “Loss of situational consciousness,”
- “Frequent forgetfulness.”

Background

“Fatigue” is a term to describe a general feeling of tiredness or lack of energy. This happens when there is no longer motivation and no energy, the most common symptom is to be “*sleepy during our daily activities.*” And this is a common symptom of “*many medical conditions with a medium to high severity range*” such as: insomnia, arthritis, fibromyalgia, hypothyroidism or hyperthyroidism, diabetes, kidney or liver disease, chronic pulmonary obstruction, autoimmune disorders, affected mental conditions, and many more as infections caused by cold, flu and/or viruses. Precisely on the concept of “*prefatigue*” which these days is very often associated with the “*COVID-19*” and our “*immune system.*” The “*prefatigue*” gradually is developed based on three stages: “*transitional,*” “*circadian,*” and “*acute fatigue.*” Where:

- “*Transitional*” as a “*prefatigue*” result of a little sleep restriction based on extended hours of sleep restriction within a range of 1 or 2 days.
- “*Circadian*” as an “*acute prefatigue*” by reducing sleeping hours during a particular “*circadian window of our natural biological internal body clock cycle,*” which is typically affected by not sleeping the period. When they have already passed sleeping “*phase 1*” and “*phase 2*” of our “*light dreams*” where we can wake up very easily, these phases because our sleep just revitalize our “*daily tiredness,*” but if we cannot

continue sleeping to “*phase 3*” and “*phase 4*” of our necessary “*deep sleep*,” where if they sleep are wake up they show a “*deteriorated cognitive system*” known as “*inertia sleep*” with an average duration from 15 to 30 minutes. And more cognitive problems occur if we cannot not reach “*phase 5*” associated with high brain activity known as “*dreaming*” or “*REM*.” At this time, a lot of brain activation is detected to process all the day’s information in parts of the brain: “*temporary superior gyri*,” “*previous cingulate*,” “*insular cortices*,” and the “*thalamus*.”

- “*Acute fatigue*” as a “*complete acute fatigue*” due to repeated restriction of accumulated hours of not getting enough sleep over a series of days

“*Prefatigue/fatigue*” damage

This whole process begins as an affectation and destruction of the nervous system, in the medium and long term that needs to be taken care of “effects due to specific causes,” as:

- “*Mental attitude causes stress*,”
- “*Physical reaction causes anxiety*,”
- “*Nervous system damage and its consequences cause panic or fear*.”
- “*Permanent nerve affects*” that affects the best weapon our body must fight the viruses, bacteria and infections causes “*our immune system be debilitated*” and give free entry to the destructive *COVID-19*.

Dataset

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be define for assigned research team.

Recommendation

To be define for assigned research team.

7.8.10 Top Challenge research project # 10 building a Cognitive health Dashboard

Case for research

Develop a “*dashboard*” to handle the “*AI and Cognitive Computing Agents System (AI-CCAS) framework*” as shown at Fig. 7.1 by voice commands or text to give orders to process the information using different “*AI Model and query results by speech orders*”

General Objective

Develop a “*dashboard*” to handle the “*AI and Cognitive Computing Agents System (AI-CCAS) framework*” by voice commands and/or text to give orders to process the information using different “*AI Model and query results by speech orders*” as explained on this chapter and using all the methodologies explained in this book.

Specific Objectives

To be define for assigned research team.

Background

To be define for assigned research team.

Dataset

To be define for assigned research team.

Procedure

To be define for assigned research team.

Conclusions

To be define for assigned research team.

Recommendation

To be define for assigned research team.

Other Challenges research project applying “*Cognitive Learning and its relationship with neuroscience of reasoning proposed as Cognitive Learning-Reasoning (CL&R) using Cognitive Computing (CC)*” using the “*AI and Cognitive Computing Agents System (AI-CCAS) framework*,” will be posted and updated on my site <http://www.garzaulloa.org>

References

- [1] J. Garza-Ulloa, Chapter 1—Introduction to biomechatronics/biomedical engineering, Applied Biomechatronics using Mathematical Models, Academic Press, 2018, pp. 1–51. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00001-9>. <http://www.sciencedirect.com/science/article/pii/B9780128125946000019>. ISBN 9780128125946.
- [2] Available from: <http://www.image-net.org/>, 11/09/2020.
- [3] Available from: <https://opensource.google/projects/open-images-dataset>, 11/02/2020.
- [4] Available from: <https://research.google.com/youtube8m/>, 11/09/2020.
- [5] Available from: <https://cocodataset.org/#home>, 11/09/2020.
- [6] Available from: <https://www.cs.columbia.edu/CAVE/software/softlib/coil-100.php>, 11/09/2020.
- [7] L. McCollum, J. Karlawish, Cognitive impairment evaluation and management, Med. Clin. North. Am. 104 (5) (2020) 807–825. Available from: <https://doi.org/10.1016/j.mcna.2020.06.007>. ISSN 0025–7125, ISBN 9780323777223.
- [8] Available from: <https://www.copemanhealthcare.com/resources/the-importance-of-baseline-cognitive-assessments>, 09/18/2020.
- [9] F. Hayes-Roth, D. Waterman, D. Lenat, Building Expert Systems, Addison-Wesley, 1983. ISBN 0–201–10686-8.
- [10] J. Garza-Ulloa, Chapter 7—Case studies of applied Biomechatronics solutions based on mathematical models, Applied Biomechatronics using Mathematical Models, Academic Press, 2018, pp. 525–626. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00007-X>. <http://www.sciencedirect.com/science/article/pii/B978012812594600007X>. ISBN 9780128125946.
- [11] Available from: https://www.cdc.gov/coronavirus/2019-ncov/symptoms-testing/symptoms.html?CDC_AA_refVal=https%3A%2F%2Fhttp://www.cdc.gov%2Fcoronavirus%2F2019-ncov%2Fabout%2Fsymptoms.html, 06/19/2020.
- [12] Available from: https://www.who.int/health-topics/coronavirus#tab=tab_3, 11/05/2020.

- [13] Available from: https://absa.org/wp-content/uploads/2020/04/ABSA2020_SARS-CoV-2_Sample_type_risk_assessment.pdf, 09/18/2020.
- [14] Available from: <https://www.cdc.gov/coronavirus/2019-ncov/cases-updates/blood-bank-serosurvey.html>, 09/19/2020.
- [15] Available from: <https://www.cdc.gov/coronavirus/2019-ncov/hcp/testing-overview.html>, 09/19/2020.
- [16] N. Bhojwani, G. Meyer MD, The rise of Healthcare AI” Partners Healthcare innovation, 2019, 11/09/2020.
- [17] Available from: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3188687/#:po=0.625000>, 11/09/2020.
- [18] R.G. Stephens, J.C. Dunn, B.K. Hayes, M.L. Kalish, A test of two processes: The effect of training on deductive and inductive reasoning, *Cognition* 199 (2020) 104223. Available from: <https://doi.org/10.1016/j.cognition.2020.104223>. <http://www.sciencedirect.com/science/article/pii/S0010027720300421>. ISSN 0010–0277.
- [19] A.E. Lawson, E.S. Daniel, Inferences of clinical diagnostic reasoning and diagnostic error, *J. Biomed. Inform.* 44 (3) (2011) 402–412. Available from: <https://doi.org/10.1016/j.jbi.2010.01.003>. <http://www.sciencedirect.com/science/article/abstract/S1532046410000043>. ISSN 1532–0464.
- [20] Available from: <http://cogsys.org/app/webroot/courses/langley/aicogsys11/notes/abduct.pdf>, 1, 2020 (accessed 27.10.21).
- [21] Available from: <https://www.sciencedaily.com/releases/2020/05/200513081810.htm>, 10/28/2020.
- [22] Available from: <https://www.mayoclinic.org/diseases-conditions/parkinsons-disease/diagnosis-treatment/drc-20376062#:~:text=No%20specific%20test%20exists%20to%20diagnose%20Parkinson%27s%20disease,.and%20symptoms%2C%20and%20a%20neurological%20and%20physical%20examination,10/27/2020>.
- [23] H. Casakin, Metaphorical reasoning and design creativity: consequences for practice and education, in: E.G. Carayannis (Ed.), *Encyclopedia of Creativity, Invention, Innovation and Entrepreneurship*, Springer, New York, NY, 2013. Available from: https://doi.org/10.1007/978-1-4614-3858-8_436.
- [24] H.H. Choi, M.J. Kim, The effects of analogical and metaphorical reasoning on design thinking, *Think. Skills Creativity* 23 (2017) 29–41. Available from: <https://doi.org/10.1016/j.tsc.2016.11.004>. ISSN 1871–1871.
- [25] J.G. Carbonell, S. Minton, *Metaphor and Common-Sense Reasoning*, Carnegie-Mellon University, Pittsburgh, PA, 1983. Available from: <https://core.ac.uk/download/pdf/208087833.pdf>. 10/28/2020, 15213.
- [26] Available from: <https://www.sciencedaily.com/releases/2020/05/200513081810.htm>, 10/28/2020.
- [27] J. Garza-Ulloa, Chapter 6—Application of mathematical models in biomechanics: artificial intelligence and time-frequency analysis, in: J. Garza-Ulloa (Ed.), *Applied Biomechanics using Mathematical Models*, Academic Press, 2018, pp. 373–524. Available from: <https://doi.org/10.1016/B978-0-12-812594-6.00006-8>. ISBN 9780128125946.
- [28] C. Quek, R.W. Zhou, POPFNN-AAR(S): a pseudo outer-product based fuzzy neural network, *IEEE Trans. Systems, Man, Cybernetics, Part. B* 29 (6) (1999) 859–870.
- [29] K.K. Ang, C. Quek, M. Pasquier, POPFNN-CRI(S): pseudo outer product based fuzzy neural network using the compositional rule of inference and singleton fuzzifier, *IEEE Trans. Systems, Man, Cybernetics, Part. B* 33 (6) (2003) 838–849.
- [30] H. Das, B. Naik, H.S. Behera, Medical disease analysis using Neuro-Fuzzy with feature extraction model for classification, *Inform. Med. Unlocked* 18 (2020) 100288. Available from: <https://doi.org/10.1016/j.imu.2019.100288>.
- [31] D.C. Krawczyk, Chapter 3—The neuroscience of reasoning, in: D.C. Krawczyk (Ed.), *Reasoning*, Academic Press, 2018, pp. 41–69. Available from: <https://doi.org/10.1016/B978-0-12-809285-9.00003-X>. ISBN 9780128092859.
- [32] J. Prado, A. Chadha, J.R. Booth, The brain network for deductive reasoning: a quantitative *meta-analysis* of 28 neuroimaging studies, *J. Cognit. Neurosci.* 23 (11) (2011) 3483–3497.
- [33] O.G. Sani, H. Abbaspourazad, Y.T. Wong, et al., Modeling behaviorally relevant neural dynamics enabled by preferential subspace identification, *Nat. Neurosci.* (2020). Available from: <https://doi.org/10.1038/s41593-020-00733-0>.
- [34] V. Goel, Cognitive neuroscience of deductive reasoning, in: K.J. Holyoak, R.G. Morrison (Eds.), *The Cambridge Handbook of Thinking and Reasoning*, Cambridge University Press, 2005.
- [35] Available from: <https://research-methodology.net/research-methodology/research-approach/abductive-reasoning-abductive-approach/>, 10/20/2020.
- [36] A. Bryman, E. Bell, *Business Research Methods*, 4th (ed.), Oxford University Press, 2015.
- [37] Available from: <https://www.parkinson.org/pd-library/books/Mood-A-Mind-Guide-to-Parkinsons-Disease>, 09/28/2020.
- [38] J. Garza-Ulloa, Theory of stress-anxiety-sleep disorders-neural damage cyclic chain and the progression of Parkinson’s 2019, disease, *Am. J. Biomed. Sci. & Res.* 6 (3) (2019). Available from: <https://doi.org/10.34297/AJBSR.2019.06.001021>. AJBSR.MS.ID.001021.
- [39] Available from: <https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html>, 09/28/2020.
- [40] Available from: <https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html#lexicon>, 11/10/2020.
- [41] Available from: <https://www.sciencedirect.com/topics/medicine-and-dentistry/event-related-functional-magnetic-resonance-imaging>, 11/10/2020.
- [42] R. Nania, This Is What the Coronavirus Can Do to Your Brain, Important Progress in Neural NLP: Modeling, Learning, and Reasoning, AARP, May 7, 2020. Available from: <https://www.aarp.org/health/conditions-treatments/info-2020/covid-19-brain-symptoms.html>, 10/13/2020.
- [43] Available from: https://www.researchgate.net/publication/353196390_Why_COVID-19_Long_Haulers, 1, 2021 (accessed 25.10.21).
- [44] W.H. Wolberg, O.L. Mangasarian, Multisurface method of pattern separation for medical diagnosis applied to breast cytology, *Proc. Natl Acad. Sci.* 87 (1990) 9193–9196. Available from: <http://www.pnas.org/content/87/23/9193.full.pdf>.
- [45] Available from: <http://mlearn.ics.uci.edu/MLRepository.html>, 11/10/2020.
- [46] Y. Zhang, S.S. Yuan, B.A. Eigel, H. Li, L.-A. Lin, W.W.B. Wang, Bayesian hierarchical model for safety signal detection in multiple clinical trials, *Contemporary Clin. Trials* (2020) 106183. Available from: <https://doi.org/10.1016/j.cct.2020.106183>. <http://www.sciencedirect.com/science/article/pii/S1551714420302615>. ISSN 1551–7144.
- [47] J.B. Tenenbaum, T.L. Griffiths, C. Kemp, Theory-based Bayesian models of inductive learning and reasoning, *Trends Cognit. Sci.*

- 10 (7) (2006) 309–318. Available from: <https://doi.org/10.1016/j.tics.2006.05.009>. ISSN 1364–6613.
- [48] C.F. Aliferis, G.F. Cooper, An evaluation of an algorithm for inductive learning of Bayesian belief networks using simulated data sets, in: R.L. de Mantaras, D. Poole (Eds.), *Uncertainty Proceedings*, Morgan Kaufmann, 1994, pp. 8–14. Available from: <https://doi.org/10.1016/B978-1-55860-332-5.50006-7>. ISBN 9781558603325.
- [49] X. Jia, P. Liang, L. Shi, D. Wang, K. Li, Prefrontal and parietal activity is modulated by the rule complexity of inductive reasoning and can be predicted by a cognitive model, *Neuropsychologia* 66 (2015) 67–74. Available from: <https://doi.org/10.1016/j.neuropsychologia.2014.10.015>. ISSN 0028–3932.
- [50] J. Buckley, N. Seery, D. Canty, L. Gumaelius, Visualization, inductive reasoning, and memory span as components of fluid intelligence: implications for technology education, *Int. J. Educ. Res.* 90 (2018) 64–77. Available from: <https://doi.org/10.1016/j.ijer.2018.05.007>. <http://www.sciencedirect.com/science/article/pii/S0883035517317561>. ISSN 0883–0355.
- [51] J. Garza-Ulloa, Update on Parkinson’s disease, *Am. J. Biomed. Sci. Res.* 2 (6) (2019). Available from: <https://doi.org/10.34297/AJBSR.2019.02.000614>. AJBSR.MS.ID.000614.
- [52] Available from: <https://www.mayoclinic.org/diseases-conditions/parkinsons-disease/expert-answers/parkinsonism/faq-20058490>
- [53] Available from: <https://www.parkinson.org/sites/default/files/attachments/Parkinsons-Disease-versus-Parkinsonisms.pdf>, 20/30/2020.
- [54] Available from: <https://study.com/academy/exam/topic/behavioral-cognitive-approaches.html>, 10/22/2020.
- [55] Book: mybraintest, Cognitive Screening and Testing Tools: Computerized Cognitive Assessment Products: Neurocognitive Screening for ADHD, Memory Loss, Alzheimer’s, Dementia, Traumatic Brain Injury, ISBN: 0615836011, ISBN13: 9780615836010.
- [56] S. Wilhelm, H. Weingarden, J.L. Greenberg, T.H. McCoy, I. Ladis, B.J. Summers, A. Matic, O. Harrison, Development and Pilot Testing of a Cognitive-Behavioral Therapy Digital Service for Body Dysmorphic Disorder, *Behav. Ther.* 51 (1) (2020) 15–26. Available from: <https://doi.org/10.1016/j.beth.2019.03.007>. <http://www.sciencedirect.com/science/article/pii/S0005789419300966>. ISSN 0005–7894.
- [57] S. Fatmawati, S. Palutturi, R. Amiruddin, M. Syafar, Development of cognitive behavior therapy apps application on depression management in patients of diabetes mellitus type II, *Enfermería Clínica* 30 (5) (2020) 21–27. Available from: <https://doi.org/10.1016/j.enfcli.2019.11.017>. ISSN 1130–8621.
- [58] S.W. Stirman, C.A. Gutner, J. Gamarra, M. Suvak, D. Vogt, C. Johnson, J.S. Wachen, K.A. Dondanville C.O.L., J.S. Yarvis, J. Mintz, A. Peterson, S. Young-McCaughan, P.A. Resick, A novel approach to the assessment of fidelity to a cognitive behavioral therapy for PTSD using clinical worksheets: a proof of concept with cognitive processing therapy, *Behav. Ther.* (2020). Available from: <https://doi.org/10.1016/j.beth.2020.08.005>. ISSN 0005–7894.
- [59] M. Meyer, S. Colnat-Coulbois, S. Frismand, P. Vidailhet, P.-M. Llorca, E. Spitz, R. Schwan, Parkinson’s disease and bilateral subthalamic nuclei deep brain stimulation: beneficial effects of preoperative cognitive restructuring therapy on postoperative social adjustment, *World Neurosurg.* 145 (2021) 282–289. Available from: <https://doi.org/10.1016/j.wneu.2020.09.128>. <http://www.sciencedirect.com/science/article/pii/S1878875020321379>. ISSN 1878–8750.
- [60] Q. Zhang, X. Yang, H. Song, Y. Jin, Cognitive behavioral therapy for depression and anxiety of Parkinson’s disease: A systematic review and meta-analysis, *Complementary Therapies Clin. Pract.* 39 (2020) 101111. Available from: <https://doi.org/10.1016/j.ctcp.2020.101111>. <http://www.sciencedirect.com/science/article/pii/S174438811930965X>. ISSN 1744–3881.

Further Reading

Available from <http://cogsys.org/app/webroot/courses/langley/aicogsys11/notes/abduct.pdf>, 10/26/2020.

Index

Note: Page numbers followed by “b,” “f,” and “t” refer to boxes, figures, and tables, respectively.

A

Abductive reasoning, 631–634. *See also*
Inductive reasoning
using AI evaluation tests, 669
causal arguments, 632–634
for medical diagnosis, 634
Acetylcholine (Ach), 42
Actions Generation, 611, 627
AdaBoost, 187
Adrenaline. *See* Epinephrine (Epi)
Adverse event prediction, 628
Advertising process, 10
Agriculture processes, 10
Air displacement plethysmography (ADP), 347
Alibaba cloud, 77
Aliyun. *See* Alibaba cloud
AlphaGo program, 6
Amazon Web Services (AWS), 77
Amygdala, 45
Analyze Neurologic opinion words, 671–672
Angina, 207
Ant colony optimization (ACO), 116
Anterior-posterior Ground Reaction forces
(apGRF), 389–390
Anthropology studies, 27, 47–48
Application Program Interface (API), 13, 60,
77–78
Applied Biomedical Engineering using
Artificial Intelligence and Cognitive
Models, challenge research for,
667–674
Arcuate fasciculus, 43
Area Under the Curve (AUC), 14
Artificial Continue Intelligence, 627
Artificial General Intelligence (AGI), 6
Artificial intelligence (AI), 1, 4–10, 76, 113,
335, 509
AI-cognitive human model, 29
algorithms, 2–3
applications of evolutionary algorithms with,
170–171
and BME, 2–3, 113–114
in medical education, 3–4
cloud service and, 76–77
cloud service providers and, 76–77
Cognitive Computing applying AI
technologies, 29–30
evolution, 7f
in healthcare, 2–3
IBM Watson Studio for, 134–169

in industries, 10
MATLAB analysis and optimization of
2D data, 124–126
3D data, 126–134
models, 31–35
optimization in BME, 114–115
evolutionary algorithms for, 115–134
genetic algorithm for, 120–121
technology evolution, 8–9
Turing Machine, 4–5
Turing test, 4–5
2D and 3D data in biomedical engineering,
121–124
types based on capabilities, 5–7
types based on functionality, 7–8
Artificial intelligence and Cognitive
Computing Agents System (AI-CCAS),
28f, 609–612, 610f, 627
architecture framework of, 9, 31
Artificial Intelligence–Machine
Learning–Deep Learning–Cognitive
Computing (AI-ML-DL-CC), 34
Artificial Narrow Intelligence (ANI).
See Narrow AI
Artificial Neural Networks (ANN), 8–9, 13,
17f, 184, 335, 336f, 509
activation functions, 337–338
deep learning based on, 335–336
backpropagation neural networks, 185
family models, 184–185
feed forward neural network family models,
184
types of, 17–18
Artificial neurons, 17
Artificial Super Intelligence (ASI). *See* Super
AI
Asset demand prediction, 628
Association Rules, 13, 178–179
Associative structures method, 34
Attention deficit hyperactivity disorder
(ADHD), 32
Attention networks (AN), 27, 603–604, 603f
Attention-deficit hyperactivity disorder
(ADHD), 604
Audio toolbox, 51
Auto Classifier, 187
Auto Cluster, 188
Auto Encoder (AE), 20–22, 336, 411–413,
412f, 414f
Auto Numeric, 188

Automated reasoning, 3–4
Automotive industry, 10

B

Backpropagation (BP), 339
algorithm, 345–346
Backpropagation neural networks, 20–22, 20f,
185, 411–442
Auto Encoder, 411–413
Deconvolutional network, 438
Deep Convolution Network or ConvNet,
437–438
Deep Convolutional Inverse Graphics
Network, 439
Deep Residual Network, 440–442
Denoising Auto Encoder, 414–415
Generative Adversarial Network, 439–440
Sparse Auto Encoder and stacked auto
encoders, 415–437
Variational Auto Encoder, 413–414
Backward propagation, 339
Backward reasoning, 612
Bag-of-Words NLP method, 49–50
Bagging, 186
Barrett’s esophagus analysis (BE analysis), 563
Basal ganglia motor loop, 45
Bayes Net, 188
Bayesian Deep Belief Network (BDBN),
592–593
Bayesian regularization backpropagation, 411
BE analysis. *See* Barrett’s esophagus analysis
(BE analysis)
Behavioral learnings, 46
Benign cancer, 415
Bernoulli Naive Bayes models, 181
Bigram model, 49f
Binary classification, 179
Bioelectrical impedance analysis (BIA), 347
Bioinformatics, 3
Biology studies, 27, 46–47
Biomedical engineering (BME), 1–4, 113,
177–178, 336
AI and, 2–3, 113–114
in medical education, 3–4
AI optimization in, 114–115
evolutionary algorithms for, 115–134
2D and 3D data in, 121–124
Biomedical instruments (bioinstruments), 29, 610
Biomedical ontologies, 3
Biomolecular computers, 35

- Blending, 186
 - Body Fat Percentage (BFP), 347
 - Body fat scales, 347
 - Body Mass Index (BMI), 346
 - Boltzmann Machine (BM), 22–25, 509, 561–562, 561*f*
 - restricted Boltzmann Machine model to reconstruct noisy chest X-ray images, 563–569
 - Bottleneck, 412
 - Brain, 40–41
 - Brain tumors, 33, 113
 - Brain–computer interfaces (BCI), 571
 - Breast cancer, 113
 - ML model, 282–309
 - Breast density, 461
 - Breast Imaging-Reporting and Data System (BIRADS), 461–462
 - Breast tumor
 - grading, 415–416
 - stage, 416
 - Building management, 10
 - Business intelligence (BI), 627
 - in healthcare, 627–628
- C**
- C5.0, 188
 - Calcifications, 461
 - Capsule Networks (CapsNet), 26–27, 601–603, 602*f*
 - Cardiology, 171
 - Care pathway identification, 628
 - Cellular modification theory, 47
 - Central nervous system (CNS), 1–2, 40
 - Central Processing Unit (CPU), 583
 - Cerebral palsy (CP), 32
 - Cerebral peduncle, 43
 - Cervical spine X-ray, 482
 - Chemical activity, 41–42
 - Chi-squared Automatic Interaction Detection (CHAID), 188
 - Circumference measurements, 347
 - Classical conditioning, 46
 - Classification, 13, 176–177, 179–180
 - Classification and Regression Tree (C&R Tree), 188
 - “Classify status of neurologic disease patients analyzing their images, movements in real time video, and speech”, 672
 - Clinical trial matching, 628
 - Cloud computing, 76–77
 - Cloud services, 60
 - and AI, 76–77
 - cloud service providers and, 76–77
 - Clustering, 179
 - Coarse decision tree, 182–183
 - Coarse Gaussian SVM, 183–184
 - Coarse kNN, 181–182
 - Coefficient of determination. *See* R-squared error (R^2)
 - Cognition
 - dopamine pathways and, 44–45, 45*f*
 - neural pathway and, 43–44
 - neurons and, 41–43
 - Cognitive apps, 77–110
 - Cognitive Behavioral Therapy (CBT), 673
 - Cognitive Computing (CC), 1–2, 9, 27–29, 110, 114, 609
 - applying AI technologies, 29–30
 - Cognitive Computing Agents Systems (CCAS), 28–29
 - Cognitive detection of human-like abilities, 29
 - Cognitive evaluations, 628
 - Cognitive health Dashboard, 674
 - Cognitive human model, 31
 - Cognitive learning, 46, 629
 - Cognitive learning and reasoning models (CL&R), 2, 9, 609, 629, 629*f*
 - Actions Generation, 627
 - AI-CCAS, 609–612
 - Applied Biomedical Engineering using Artificial Intelligence and Cognitive Models, 667–674
 - business intelligence in healthcare, 627–628
 - Inference engine and research example, 612–627
 - learning and reasoning relationship of biomedical engineering, cognitive science, and computer science, 629–640
 - research example applying AI-CCAS framework, 640–667
 - Cognitive learning and relationship with neuroscience of reasoning, 640
 - Cognitive science (CoSi), 1, 27–35, 45–48, 46*f*, 110, 629
 - Cognitive Computing, 28–29
 - applying AI technologies, 29–30
 - cognitive detection of human-like abilities, 29
 - cognitive model obtainment, 30–31
 - inference engine, 31
 - signal recognition instruments, 29
 - Cognitive technology. *See* Cognitive computing (CC)
 - Cognitive Theory Theory (CTT), 671
 - Comma-separated values (csv), 12
 - Computational intelligence (CI), 2–3
 - Computational statistics, 11
 - Computed tomography (CT), 170, 461
 - Computer personal assistant, 48
 - Computer science (CS), 1
 - “Computer Science through Artificial Intelligence”, 27
 - Computer vision, 536–537
 - toolbox, 51
 - Concrete operational stage, 41
 - Condition management, 628
 - Conditioned reinforcement. *See* Secondary reinforcement
 - Confidence, 179
 - Confusion matrix, 13
 - Connectionism technique, 34
 - Constraints function, 120–121
 - Content lookup, 583
 - Content-based addressing, 581–582
 - ConvNet (CNN), 336, 411, 437–438
 - Convolution shift, 582
 - Convolutional neural network (CNN), 510
 - Corpus callosum, 43
 - Corticospinal tract, 43–44
 - Cosine kNN, 181–182
 - Covariance matrix adaptation evolution strategy (CMA-ES), 570
 - COVID-19 lung imaging chest X-rays, 573–574
 - Cox Regression, 189
 - Craniocaudal View (CC View), 443
 - Cross-validation, 13
 - Cubic kNN, 181–182
 - Customer services processes, 10
 - Cutoff selection, 117
 - Cyber-Physical Systems (CPS), 10
 - Cyst, 461
- D**
- Data collection, 12, 175
 - Data mining, 1, 11
 - Data preparation and exploration, 12, 175
 - Data schema, 12
 - Data vectorization in NLP, 49–50
 - Data-reduction techniques, 189
 - Database, 11
 - Death density function, 178
 - Decision boundary, 335
 - Decision Trees (DT), 13, 182
 - family models, 182–183
 - for supervised learning, 182–183
 - Decoder, 412, 602
 - Deconvolution layer, 438
 - Deconvolutional Network (DN), 20–22, 336, 411, 438
 - Deconvs. *See* Deconvolutional network (DN)
 - Deductive reasoning, 48, 630, 654*f*
 - Disjunction Elimination, 630
 - Modus Ponens, 630
 - Modus Tollens, 630
 - Deep Belief Network (DBN), 26, 592–601, 593*f*, 595*f*
 - Deep Blue IBM’s chess-playing supercomputer, 7
 - Deep Convolution Network (DCN), 20–22, 336, 411, 437–438, 437*f*
 - Deep Convolutional Inverse Graphics Network (DCIGN), 20–22, 336, 411, 439, 440*f*
 - Deep feed forward network (DFF), 18–19
 - Deep Feed forward Network (DFFN), 338–339
 - Deep learning (DL), 1, 8–9, 16–27, 30, 114, 335. *See also* Pretrained deep learning networks
 - artificial neural networks, 17–18
 - backpropagation neural network, 20–22, 20*f*
 - evolutionary deep neural networks, 26–27, 26*f*
 - feed forward neural network, 18–20, 19*f*
 - machine learning vs., 16–17
 - memory augmented neural networks, 25, 25*f*
 - models evolution applied to biomedical engineering, 509
 - Evolutionary Deep Neural Networks, 601–604
 - MANN, 581–592

Modular Neural Networks, 592–601
 recurrent neural networks, 509–580
 models principles applied to biomedical engineering
 backpropagation neural networks types, 411–442
 deep learning based on artificial neural networks, 335–336
 feed forward neural networks types, 336–346
 shallow neural network, 346–411
 transfer learning from pretrained deep learning networks, 442–505
 modular neural networks, 25–26, 26f
 recurrent neural networks, 22–24, 23f
 toolbox, 51
 Deep Residual Network (DRN), 20–22, 336, 411, 440–442, 441f
 Deep ResNet, 336, 411
 Demyelinating disease. *See* Multiple sclerosis (MS)
 Denoising Auto Encoder (DAE), 20–22, 336, 411, 414–415
 Density-Based Spatial clustering (DBScan), 12–13
 Deployment model, 16
 Detect hypoglycemia, 171
 Diabetes analysis, SPSS Modeler flow for, 138–169
 Diabetes Mellitus (DM), 371
 Diabetes ML model, 191
 Differentiable Neural Computers (DNC), 25, 583–592, 583f
 Differential evolution (DE), 116
 Digital mammography, 443
 Direct search algorithm, 125
 Discriminant analysis, 189
 for face recognition, 185
 family models, 185
 for medical applications, 185
 in robots, 185
 Discriminator, 440
 Disease pattern identification, 628
 Disjunction Elimination, 630
 Dopamine (DA), 42
 Dopamine pathways and cognition, 44–45, 45f
 Dorsal column-medial lemniscus pathway (DCML), 43
 Drug discovery, 3
 Drug regimen selection, 628
 DSP system toolbox, 51
 Dual-energy X-ray absorptiometry (DEXA), 347
 Dynamic memory allocation, 583–584

E
 Echo State Network (ESN), 22–25, 509, 570–580, 571f
 Education process, 47
 Electrical activity, 41
 Electrocardiograph (ECG), 29, 121–122
 Electroencephalogram (EEG), 29, 170, 571, 610
 Electromyography (EMG), 29, 121–122, 610

Electrooculography (EOG), 29
 Elitist selection, 117
 Encoder, 412, 601–602
 Endocrinology, 171
 Endorphins, 43
 Energy-Based Models (EBM), 561
 Ensemble classifiers family models, 186–187
 Epinephrine (Epi), 43
 Euclidean distance, 182
 Evolutionary algorithms
 for AI, 115–134
 typical algorithms, 116–118, 119f
 Evolutionary Deep Neural Networks, 26–27, 26f, 601–604
 attention networks, 603–604, 603f
 CapsNet, 601–603
 Evolvable Neural Turing Machine (ENTM), 582
 Exhaustive data, 176
 Expectation maximization, 13
 Explained variance, 15
 Exploratory data analysis (EDA), 11
 Extreme Learning Machine (ELM), 18–20, 336, 343–346, 344f, 571

F
 False positive rate (FPR), 14
 Fast Regional-Convolutional Neural Network (Fast R-CNN), 537
 Fasting blood sugar test, 191
 Feature engineering, 12, 175, 335
 in NLP, 50
 Feature Selection, 189
 Feed forward Neural Network (FFNN), 18–20, 19f, 338–339
 to Analyze “Human Body Fat”, 346–370
 case for research, 346
 dataset, 347
 general objective, 346
 procedure, 347–370
 specific objectives, 346
 family models, 184
 types, 336–346
 perceptron, 337–338
 Fetal electrocardiogram (fECG), 592–593
 FibeR-CNN, 537
 Fibroadenomas, 461
 Film-screen mammography, 443
 Fine decision tree, 182–183
 Fine Gaussian SVM, 183–184
 Fine kNN, 181–182
 Fine Needle Aspiration (FNA), 416
 Fitness function, 120
 Fitness proportionate selection, 117
 Flu, 342
 Formal operational stage, 41
 Forward propagation, 339
 Forward reasoning, 612
 Frequency of words (FW), 49
 Functional Magnetic Resonance Imaging (fMRI), 170, 461
 Fuzzy inference systems (FIS), 51
 Fuzzy logic toolbox, 51

G
 Gamma-aminobutyric acid (GABA), 42
 Gated recurrent unit networks (GRU networks), 22–25, 509, 512–536, 513f
 Gaussian Naive Bayes models, 181
 General linear model (GenLin), 188
 Generalized linear mixed model (GLMM), 188
 Generative Adversarial Network (GAN), 20–22, 336, 411, 439–440, 441f
 Generator, 440
 Genes, 117
 Genetic algorithms (GA), 8, 115, 118, 125
 for AI optimization, 120–121
 Genetic programming (GP), 115
 GenLin. *See* General linear model (GenLin)
 Genomics, 3
 Gestalt theory of learning. *See* Insight learning
 Gestational diabetes, 191
 Global search algorithm, 125
 Glu. *See* Glutamate (Glu)
 Glutamate (Glu), 42–43
 Glycated hemoglobin test (A1C test), 191
 GoogLeNet, 461
 Gradient Descent method, 411
 Graphics processing units (GPU), 17
 Greedy learning algorithm, 592
 Ground Reaction Force (GRF), 121–122, 389–390

H
 Hamming distance, 182
 Happiness hormone. *See* Serotonin (5-HT)
 Hazard function, 178
 Healthcare using artificial Intelligence, 2
 Heart attack, 207
 Heart disease ML model, 207–245
 Hierarchical clustering, 179
 High-performance language, 12
 Hippocampus, 47
 Holdout method, 176
 Hopfield Network (HN), 22–25, 509, 554–561, 554f
 model to reconstruct noisy chest X-ray images, 555–561
 Human cognition, 110
 Human diseases, 9
 Human disorders, 9
 Human illness, 9
 Human resources processes, 10
 Human voice cognitive analysis for cognitive services as voice therapy, 672
 Hydrodensitometry, 347
 Hyperparameter, 15–16
 Hyperparameter optimization. *See* Tuning the model

I
 IBM cloud, 77–110
 to create APIs, 78–110
 to create APIs NLP, 80r
 for NLP, 78

IBM ML Solution, 187–333
 SPSS Modeler flows > modeling,
 187–189
 SPSS Modeler flows > output, 189–333

IBM Watson, 77–110
 SPSS, 187–333
 Studio for AI, 134–169

Identify Function, 414
 Image Analysis, 628
 Image Generation, 31, 611
 Image processing toolbox, 51
 Import functions, 56
 Impurity measurement function, 183
 Inductive learning, 48

Inductive reasoning, 630–631
 AI evaluation test, 667–669
 analogical arguments, 631
 causal arguments, 631
 generalized arguments, 631
 predictive arguments, 631
 statistical arguments, 631

Industrial Internet of Things (IIOT), 10
 Industry 4.0*, 10–11
 Inference Engine, 31, 611
 and research example, 612–627
 Infinity Restricted Boltzmann Machines
 (iRBMs), 563

Influenza, 342
 Information extraction, 48
 Information retrieval, 48
 Information technology (IT), 77
 Infrastructure-as-a-Service (IaaS), 76
 Input vector, 340
 Insight learning, 46
 Interactive voice response (IVR), 48
 Interconnection strengths, 18
 Internet of Things (IoT), 10, 76
 Interpolation, 582

Intracranial neoplasm. *See* Brain tumors
 Inverse convolution model, 21–22
 Inverse document frequency (IDF), 50
 Izhikevich neuronal mathematical model,
 572–573

J

JavaScript Object Notation (json), 12

K

K-fold cross-validation, 176
k-means clustering, 12
k-modes clustering, 12
k-Nearest neighbor (kNN), 181
 analysis, 189
 family models, 181–182
 for supervised learning, 181–182
k-prototypes, 12
 Kernel Fisher discriminant analysis, 19
 Kidney disease ML model, 245–282
 Knapsack-problem, 118
 Knowledge discovery and data mining, 30
 Knowledge Discovery in Database (KDD), 11,
 178
 Knowledge representation, 4

Knowledge Storage AI Database (KSAIDB),
 28–29, 611, 635
 Korhonen network (KEN), 22–25, 509, 580

L

Language translation, 48
 Laser-Induced Breakdown Spectroscopy
 (LIBS), 562–563
 Law of contradiction, 5
 Law of principle of identity, 5
 Law to exclude middle, 5
 Leaky integrate-and-fire neurons (LIF), 594
 Learning, 4
 process, 17–18
 reasoning relationship of biomedical
 engineering, cognitive science, and
 computer science, and, 629–640
 Leave-one-out cross-validation, 176
 Leave-P-out cross-validation, 176
 Lemmatizing method, 49
 Levenberg–Marquardt backpropagation, 411
 Lexical analyzer method, 49
 Lift, 179
 Limited Memory type AI systems, 7
 Linear binary classifier, 18
 Linear Discriminant Analysis (LDA), 185
 Linear programming (LP), 115
 Linear regression, 180
 Linear support vector machine (LSVM),
 183–184, 189
 Linguistics, 47–48
 studies, 27
 Liquid State Machine (LSM), 22–25, 509,
 569–570, 569*f*
 of node-neurons from “Spiking Neural
 Networks”, 572–580
 Living biohybrid systems, 35
 Location-based addressing, 582
 Logic AND function, 5
 Logic OR function, 5
 Logistic regression, 13, 180, 189
 classifier, 186
 family models, 186
 Long-standing approach, 3
 Long-term potentiation (LTP), 47
 Long/short-term memory (LSTM), 22–25, 509,
 511–512, 511*f*
 to classify videos about human body
 movements and detect human falls,
 513–536
 Loss calculation, 339
 Loss function, 15–16
 Loss of consciousness (LOC), 33
 Low-dose X-rays, 461

M

Machine learning (ML), 1, 4, 6, 8, 11–16, 30,
 114, 175, 335
 artificial neural network family models,
 184–185
 choosing best ML model, 175–179
 Association Rules, 178–179
 reinforcement learning, 177–178

supervised learning, 176–177
 survival models, 178
 unsupervised learning, 176
 clusters, classification, and regression
 models, 179–180
 data collection, 12
 data preparation and exploration, 12
 decision trees family models for supervised
 learning, 182–183
 deep learning vs., 16–17
 discriminant analysis family models, 185
 ensemble classifiers family models,
 186–187
 feature engineering, 12
 IBM ML Solution, 187–333
k-Nearest neighbor family models for
 supervised learning, 181–182
 logistic regression classifier, 186
 model evaluation and tuning, 13–15
 model prediction and deployment, 16
 model selection, 12–13
 for NLP, 50
 model training, 13
 Naive Bayes family models for supervised
 learning, 180–181
 steps, 11–16, 11*f*
 support vector machine family members,
 183–184

Magnetic Resonance Imaging (MRI), 170, 461
 Malignant cancer, 415
 Mamdani-type inference in MATLAB, 637
 Mammography-NT, 463*f*
 Manhattan distance, 182
 Marketing processes, 10
 Matching matrix, 13
 Mathematical optimization, 11
 Mathematical programming. *See* Mathematical
 optimization
 Mathematics, 27
 MATLAB ML solution: Statistics and Machine
 Learning Toolbox, 309
 MATLAB®
 analysis and optimization of 2D data,
 124–126
 analysis and optimization of 3D data,
 126–134
 genetic algorithm for AI optimization,
 120–121
 IBM Cloud API key for MATLAB speech to
 text, 62*t*
 for NLP, 50–76
 NLP applications with, 51–52
 NLP audio files with, 57–58
 NLP speech to text with, 60–76
 NLP text to speech using, 58–60
 NLP topic models with, 52–57
 patient with neurologic disease, 52–57
 Maximum a posteriori (MAP), 13
 Maximum Likelihood Estimation (MLE), 186
 Medial forebrain bundle (MFB), 43
 Medial-lateral Ground Reaction forces
 (mlGRF), 389–390
 Medical data mining, 3
 Medical imaging, 3

- Medical informatics, 3
 Medication administration, 628
 Medicine and healthcare, 10
 Medio Lateral Oblique (MLO), 443
 Mediodorsal thalamus, 47
 Medium decision tree, 182–183
 Medium Gaussian SVM, 183–184
 Medium kNN, 181–182
 Meiosis, 117
 Mel-frequency cepstrum coefficient (MFCC), 510
 Memetic algorithms (MA), 116
 Memory augmented neural networks (MANN), 25, 581–592, 581*f*
 DNC, 583–592
 NTM, 581–583
 Memory networks. *See* Memory augmented neural networks
 Mesocortical pathway, 44
 Mesolimbic pathway, 44–45
 Meta-reasoning, 3
 Metaphoric reasoning, 635–636
 Metaphoric reasoning for clinical diagnosis using Cognitive Learning and Reasoning applying Cognitive Computing, 669–670
 Microsoft azure, 77
 Minkowski distance, 182
 Mitosis, 117
 Model & predict functions, 56
 Model evaluation and tuning, 13–15, 175
 Model prediction and deployment, 16, 175
 Model selection, 12–13, 175
 Model training, 13, 175
 Modified Pretrained Network, 461
 Modular Neural Networks, 25–26, 26*f*, 592–601
 DBN, 592–601
 Modus Ponens, 630
 Modus Tollens, 630
 Motion Generation, 611
 Motion generation, 31
 Motor, 39
 pathways, 43
 symptoms, 39
 Multiclass classification, 179
 Multilabel classification, 179–180
 Multilayer Perceptron (MLP), 18–19, 336, 338–340
 Multinomial Naive Bayes models, 181
 Multiple sclerosis (MS), 32–33, 170
 Muscular dystrophy (MD), 32
 Muscular system, 40
 Musculoskeletal system, 40
- N**
 Naive Bayes classifiers, 180
 Naive Bayes family models for supervised learning, 180–181
 Gaussian Naive Bayes, multinomial Naive Bayes, Bernoulli Naive Bayes models, 181
 Narrow AI, 6
 Natural language generation (NLG), 47–48
 Natural Language Processing (NLP), 1, 4, 6, 30, 47–50, 48*f*, 611
 data vectorization in, 49–50
 feature engineering in, 50
 IBM cloud for, 78
 MATLAB® toolboxes solution for, 50–76
 IBM Cloud API key for MATLAB speech to text, 62*f*
 for NLP, 50–76
 NLP applications with, 51–52
 NLP audio files with, 57–58
 NLP speech to text with, 60–76
 NLP text to speech using, 58–60
 NLP topic models with, 52–57
 patient with neurologic disease, 52–57
 ML model selection for, 50
 preprocessing text for, 49
 reading dataset for, 49
 Natural language understanding (NLU), 47
 Nerves, 40–41
 Nervous system, 40
 Neural architecture search (NAS), 593
 Neural NET, 189
 Neural network
 for clustering based on “Self-Organizing Map”, 370–389
 case for research, 370
 dataset, 371
 Diabetes Mellitus, 371
 general objective, 370
 procedure, 371–389
 sEMG, 370–371
 specific objectives, 370
 computer, 34–35
 Neural pathways, 41
 and cognition, 43–44
 Neural Turing Machines (NTM), 25, 581–583
 Neuro-Fuzzy logic reasoning as cognitive reasoning, 636–639
 Neuro-Fuzzy systems (NFS), 639
 Neurologic—evaluating anxiety in neurologic diseases using Cognitive Therapy Theory using Cognitive Learning and Reasoning with Cognitive Computing, 670–671
 Neurology, 1–2, 170
 Neurons, 41
 and cognition, 41–43
 Neuroscience, 31–35
 studies, 27, 47
 Neurotransmitters, 41–43
 dopamine pathways, 45
 Nigrostriatal pathway, 45
 Nominal regression. *See* Logistic regression
 Nonexhaustive data, 176
 Nonlinear autoregressive exogenous model, 389–411
 case for research, 389
 dataset, 390
 general objective, 389
 procedure, 390–411
 specific objectives, 389
 vertical Ground Reaction Forces, 389–390
 Nonlinear programming (NP), 115
 Nonmotor symptoms, 39–41
 Nonprobabilistic model, 184
 Norepinephrine (NE), 43
 Nucleotide rearrangement theory, 47
 Nucleus accumbens, 44–45
- O**
 Observational learning, 46
 Oncology, 170–171
 Open Neural Network Exchange (ONNX), 442
 Open source language, 12
 Operant conditioning, 46
 Oracle cloud, 77
 Oral glucose tolerance test, 191
 Output layer, 340
- P**
 Parametric Rectified Linear Unit (PreLU), 337–338
 Parkinson’s disease (PD), 33
 Particle swarm algorithm, 125
 Particle Swarm Optimization (PSO), 115–116, 593
 Partitional clustering, 179
 Patient data flagging, 628
 Patient flow optimization, 628
 Patient intervention suggestions, 628
 Patient self-monitoring, 628
 Pattern recognition, 1
 Pavlovian conditioning. *See* Classical conditioning
 Perceptron (P), 18–19, 336–338
 ANN activation functions, 337–338
 Extreme Learning Machine, 343–346
 multilayer perceptron, 338–340
 probabilistic neural network, 342–343
 RBF network, 340–341
 Peripheral nervous system (PNS), 1–2, 40
 Phenotype, 117
 Philosophy studies, 48
 Philosophy studies, 27
 Physical therapy, 628
 Platform-as-a-Service (PaaS), 76
 Polynomial regression, 180
 Positron emission tomography (PET), 3
 Posterior column-medial lemniscus pathway (PCML), 43
 Postsynaptic neuron, 41–42
 Precision, 13–14
 Precollected data, 12
 Prefatigue/fatigue by stress and anxiety detection, 673–674
 Preoperational stage, 41
 Preprocess functions, 56–57
 Presynaptic neuron, 41–42
 Pretrained deep learning networks, 442*f*
 AI model to classify Mammograms standard views types, 443–461
 and breast abnormalities as possible breast tumor, 461–482
 custom Deep Convolutional Neural Network, 482–505
 transfer learning from, 442–505

Primary reinforcement, 177
 Principal Component Analysis (PCA), 12, 562–563
 Probabilistic generative methods, 184
 Probabilistic model, 184
 Probabilistic Neural network (PNN), 18–19, 184, 336, 342–343, 342*f*
 Probability distribution function (PDF), 19, 184, 342
 Proteomics, 3
 Psychology studies, 27, 46
 Pyramidal tracts, 43

Q
 Quadratic Discriminant Analysis (QDA), 185
 Quadratic programming (QP), 115
 Quick, unbiased, efficient statistical tree (Quest), 188

R
 R-squared error (R^2), 15
 Radial basis function network (RBF network), 18–19, 336, 340–341, 340*f*
 Radiation plan design, 628
 Radiology, 170
 Radiotherapy, 171
 Random blood sugar test, 191
 Random decision forests, 187
 Random forests, 13, 187
 Random-access memory (RAM), 25
 Rational agents, 5
 theory, 5
 Reactive Machines type systems, 7
 Reasoning, 4
 Recall, 14
 Receiver operating characteristic (ROC), 14
 Recommendation systems, 13
 Reconstruction loss, 412
 Rectified Linear Unit known (RELU), 337–338
 Recurrent convolutional neural networks (RCNN), 22–25, 509, 536–537, 536*f*
 Recurrent neural networks (RNN), 22–24, 23*f*, 509–580. *See also* Artificial Neural Networks (ANN)
 Boltzmann Machine, 561–562
 echo state network, 570–580
 gated recurrent unit networks, 512–536
 Hopfield network, 554–561
 Korhonen Network, 580
 Liquid State Machine, 569–570
 long/short-term memory, 511–512
 recurrent convolutional neural networks, 536
 regional-convolutional neural network object detection in AI models, 536–554
 Restricted Boltzmann Machine, 562–569
 vanilla, 509–510, 510*f*
 Regional-convolutional neural network (RCNN), 23. *See also* Shallow neural network
 object detection
 in AI models, 536–554
 of breast tumor in mammogram, 537–554

Regression, 13, 177, 180
 Reinforcement learning (RL), 13, 177–178
 algorithms, 6–7
 Remove punctuation, 49
 Remove stop-words, 49
 Respondent conditioning. *See* Classical conditioning
 Restricted Boltzmann Machines (RBMs), 22–25, 509, 562–569, 562*f*, 592
 neurons, 340
 Retinohypothalamic tract, 43
 Ribonucleic acid (RNA), 47
 Ridge regression, 180
 Right posterior parietal cortex (r-PPC), 604
 Risk prediction, 628
 Robotic-assisted surgery, 628
 Roulette-wheel selection, 117
 RUSBoost, 187

S
 Salesforce, 77
 SARS-CoV-2, 584–585
 Scar tissues, 461
 Secondary motor, 39
 Secondary reinforcement, 177
 Self-Awareness type of AI systems, 8
 Self-correction, 4
 Self-learning response model (SLRM), 189, 580
 Self-Organizing Map (SOM), 370
 Input Planes, 370
 Neighbor Distance, 370
 Sample Hits, 370
 Topology, 370
 Semantic types, 12
 Semantic Web, 30
 Sensorimotor stage, 41
 Sensory pathways, 43
 Sentiment analysis, 30, 48
 Serotonin (5-HT), 42
 Shallow neural network, 346–411
 Feed Forward Neural Network to Analyze “Human Body Fat”, 346–370
 neural network for clustering based on “Self-Organizing Map”, 370–389
 nonlinear autoregressive exogenous model, 389–411
 Sharpening, 582
 Siegert neuron, 594
 Sigmoid/logistic function, 186
 Signal recognition instruments, 29
 Single-Layer Feed Forward Neural Network (SLFN), 345–346
 Single-photon emission computed tomography (SPECT), 3
 Skeletal system, 40
 Skinfold measurements, 347
 Small white specks, 461
 Softmax function, 26
 Software-as-a-Service (SaaS), 76
 Spam filters, 48
 Sparse Auto Encoder (SAE), 20–22, 185, 336, 411, 415–437
 patterns recognition and classification of “breast cancer”, 415–437

case for research, 415
 dataset, 416
 general objective, 415
 procedure, 416–437
 specific objectives, 415
 Spatiotemporal prediction (STP), 189
 Speech capture, 29
 Speech Generation, 47–48, 611
 Speech processing, 30
 Speech recognition, 29, 48, 610
 Speech to text, 29
 Spiking deep belief network (SDBN), 594
 Spiking neural networks (SNNs), 24, 569
 Spinal cord, 40–41
 Spinal cord injury (SCI), 33–34
 Spinothalamic tract, 43–44
 SPSS Modeler flows > modeling, 187–189
 SPSS Modeler flows > output, 189–333
 Stacked auto encoders, 21, 415–437
 Stacking, 186
 Standard views, 443
 Statistical computing. *See* Computational statistics
 Statistical Package for Social Sciences Modeler flows (SPSS Modeler flows), 136
 IBM SPSS Modeler flow, 136–137
 for general dataset analysis, 138–169
 Statistics, 11
 Statistics and machine learning toolbox, 51
 Stochastic Gradient Variational Bayes algorithm (SGVB algorithm), 22, 439
 Stratified K-fold cross-validation, 176
 String functions, 57
 Stroke, 207
 Strong AI. *See* Super AI
 Subspace discriminant, 187
 Subspace kNN, 187
 Sugeno-type inference in MATLAB, 637–639
 Super AI, 6
 Supervised learning, 13, 176–177, 184
 Supplementary views, 444
 Supply chain processes, 10
 Support, 179
 Support Vector Machine (SVM), 13, 183, 189
 family members, 183–184
 family models, 183–184
 Support vector regression, 180
 Surface Electromyography (sEMG), 370–371
 Survival analysis, 178
 Survival function, 178
 Survival models, 13, 178
 Swarm Algorithms, 8
 Synapse, 42

T
 Telehealth, 628
 Temporal causal model (TCM), 188
 Temporary memory linkage, 583
 Term frequency (TF), 50
 Term frequency inverse document frequency (TF-IDF), 50
 Text analytics toolbox, 51
 Text simplification, 48
 Text summarization, 48
 “Theory of Mind” type AI systems, 7–8

- 3D
 - body scanners, 347
 - data
 - in biomedical engineering, 121–124
 - MATLAB analysis and optimization of 3D data, 126–134
 - Token, 49*t*
 - Tokenization, 49
 - Tokenizer, 50
 - Total number of terms (TN), 49
 - Traditional search methods, 8
 - Trail-and-error learning, 46
 - Transcranial direct current stimulation (tDCS), 604
 - Transfer learning, 34
 - from pretrained deep learning networks, 442–505
 - Transposed convolutional layer.
 - See* Deconvolution layer
 - Traumatic brain injury (TBI), 33
 - Trial-and-error approach, 46
 - True positive rate (TPR), 14
 - Tuberoinfundibular pathway, 45
 - Tumor, 461
 - Tuning the model, 15–16
 - Turing Machine (TM), 4–5, 584
 - simulation using recursive function, 584–592
 - Turing test, 4–5
 - Tutorial IBM Watson SPSS Modeler Flow
 - breast cancer ML model, 282–309
 - diabetes ML model, 190–207
 - heart disease ML model, 207–245
 - kidney disease ML model, 245–282
 - Statistics and Machine Learning Toolbox, 309–333
 - 2D data
 - in biomedical engineering, 121–124
 - MATLAB analysis and optimization of, 124–126
 - Type 1 diabetes, 191
 - Type 1 synapse, 17–18
 - Type 2 diabetes, 191
 - Type A flu, 342
 - Type B flu, 342
 - Type C flu, 342
- U**
- Ultrasound imaging, 170, 461
 - Unconditional reinforcement. *See* Primary reinforcement
 - Underwater weighing, 347
 - Unpooling layer, 438
 - Unsupervised learning, 12–13, 176, 184
- V**
- Validation, 176
 - Vanilla, 509–510, 510*f*
 - Variational Auto Encoder (VAE), 20–22, 336, 411, 413–414
 - Ventral tegmental area (VTA), 44
 - vertical Ground Reaction Forces (vGRF), 389–390
 - Vision recognition, 29, 610
 - Visualization functions, 56
 - Visuospatial relational reasoning, 639–640
- W**
- Weak AI. *See* Narrow AI
 - Word level analyzer, 50
- Y**
- You Only Look Once (YOLO), 537

Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models

Jorge Garza-Ulloa

Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models focuses on the relationship between three different multidisciplinary branches of engineering: Biomedical Engineering, Cognitive Science, and Computer Science through Artificial Intelligence (AI) models. These models will be used to study how the nervous system and musculoskeletal system obey movement orders from the brain, as well as the mental processes of the information during cognition when injuries and neurologic diseases are present in the human body. The interaction between these three areas is studied in this book with the objective of obtaining AI models on injuries and neurologic diseases of the human body, studying diseases of the brain, spine, and the nerves that connect them with the musculoskeletal system. There are more than 600 diseases of the nervous system, including brain tumors, epilepsy, Parkinson's disease, and stroke. These diseases affect the human cognitive system that sends orders from the central nervous system through the peripheral nervous systems to do tasks using the musculoskeletal system. These actions can be detected by many Bioinstruments (Biomedical Instruments) and cognitive device data, allowing us to apply AI using Machine Learning-Deep Learning-Cognitive Computing (ML-DL-CC) models through algorithms to analyze, detect, classify, and forecast the process of various illnesses, diseases, and injuries of the human body. *Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models* provides readers with the study of injuries, illness, and neurological diseases of the human body through AI using ML-DL-CC models based on algorithms developed with MATLAB® and IBM Watson®.

Key Features

- Provides an introduction to cognitive science, cognitive computing, and human cognitive relation to help in the solution of AI Biomedical Engineering problems.
- Explains different AI, including evolutionary algorithms to emulate natural evolution, reinforced learning, Artificial Neural Network type, and cognitive learning and to obtain many AI models for Biomedical Engineering problems.
- Includes coverage of the evolution of AI through ML, DL, CC using MATLAB® programming language with many Add-On MATLAB® toolboxes and AI-based commercial products cloud services as IBM (Cognitive Computing, IBM Watson®, IBM Watson Studio®, IBM Watson Studio® Visual Recognition) and others.

Dr. Jorge Garza Ulloa is the author of the Elsevier book *Applied Biomechanics Using Mathematical Models* that provides appropriate methodologies to detect and measure diseases and injuries related to the human kinematics and kinetics, featuring mathematical models that, when applied to engineering principles and techniques in the medical field, can be used in assistive devices that work with bodily signals. The proposed book *Applied Biomedical Engineering Using Artificial Intelligence and Cognitive Models* is an evolution of concepts and techniques explained in his former book (Note: It is not necessary to be read in sequence, but they complement each other), now integrating three multidiscipline: Biomedical Engineering, Cognitive Science, and Computer Science algorithms to propose solutions for injuries and neurologic diseases based on AI modeling. Dr. Garza Ulloa research is in the development of mathematical models for Biomedical Applications, as the Mathematical Model for the Validation of the Ground Reaction Force Sensor in Human Gait Analysis, the Mathematical model to predict Transition-to-Fatigue during isometric exercise on muscles of the lower extremities, and mathematical model to be used in the Assessment and evaluation of dynamic behavior of muscles with special reference to subjects with Diabetes Mellitus.



ACADEMIC PRESS

An imprint of Elsevier

elsevier.com/books-and-journals

ISBN 978-0-12-820718-5



9 780128 207185