

HANDBOOK OF RESEARCH ON

Machine Learning Applications and Trends

Algorithms, Methods, and Techniques



Emilia Sonia Olivas, José David Martín Guerrero, Manuela Martínez Sobejano,
José Rafael Magdalena Benedito, & Antonio José Serrano López

VOLUME I

Olivas, Guerrero, Sobejano, Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques

Volume III

Information Science
REFERENCE

Volume II

Information Science
REFERENCE

Volume I

Information Science
REFERENCE

Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods, and Techniques

Emilio Soria Olivas
University of Valencia, Spain

José David Martín Guerrero
University of Valencia, Spain

Marcelino Martínez Sober
University of Valencia, Spain

Jose Rafael Magdalena Benedito
University of Valencia, Spain

Antonio José Serrano López
University of Valencia, Spain

**Information Science
REFERENCE**

INFORMATION SCIENCE REFERENCE
Hershey • New York

Director of Editorial Content: Kristin Klinger
Senior Managing Editor: Jamie Snavely
Assistant Managing Editor: Michael Brehm
Publishing Assistant: Sean Woznicki
Typesetter: Sean Woznicki
Cover Design: Lisa Tosheff
Printed at: Yurchak Printing Inc.

Published in the United States of America by

Information Science Reference (an imprint of IGI Global)
701 E. Chocolate Avenue
Hershey PA 17033
Tel: 717-533-8845
Fax: 717-533-8661
E-mail: cust@igi-global.com
Web site: <http://www.igi-global.com/reference>

Copyright © 2010 by IGI Global. All rights reserved. No part of this publication may be reproduced, stored or distributed in any form or by any means, electronic or mechanical, including photocopying, without written permission from the publisher.

Product or company names used in this set are for identification purposes only. Inclusion of the names of the products or companies does not indicate a claim of ownership by IGI Global of the trademark or registered trademark.

Library of Congress Cataloging-in-Publication Data

Handbook of research on machine learning applications and trends : algorithms, methods and techniques / Emilio Soria Olivas ... [et al.]

p. cm.

Summary: "This book investigates machine learning (ML), one of the most fruitful fields of current research, both in the proposal of new techniques and theoretic algorithms and in their application to real-life problems"--

Provided by publisher.

Includes bibliographical references and index.

ISBN 978-1-60566-766-9 (hardcover) -- ISBN 978-1-60566-767-6 (ebook) 1.

Machine learning--Congresses. 2. Machine learning--Industrial applications--Congresses. I. Soria Olivas, Emilio, 1969-

Q325.5.H36 2010

006.3'1--dc22

2009007738

British Cataloguing in Publication Data

A Cataloguing in Publication record for this book is available from the British Library.

All work contributed to this book is new, previously-unpublished material. The views expressed in this book are those of the authors, but not necessarily of the publisher.

Editorial Advisory Board

Todor Ganchev, *University of Patras, Greece*

Marina Sokolova, *CHEO Research Institute, Canada*

Roy Rada, *UMBC, USA*

Pedro J. García-Laencina, *Universidad Politécnica de Cartagena, Spain*

Vadlamani Ravi, *Institute for Development and Research in Banking Technology (IDRBT), India*

List of Reviewers

Tsan-Ming Choi, *The Hong Kong Polytechnic University, Hong Kong*

V. Ravi, *Institute for Development and Research in Banking Technology (IDRBT), India*

Roy Rada, *Department of Information Systems, UMBC, USA*

Todor Ganchev, *University of Patras, Greece*

Albert Ali Salah, *Centre for Mathematics and Computer Science (CWI), The Netherlands*

Stefano Ferilli*, *Dipartimento di Informatica – Università degli Studi di Bari, Italy*

Paul Honeine, *Institut Charles Delaunay (FRE CNRS 2848), Laboratoire de Modélisation et Sécurité des Systèmes (LM2S), France*

Adam E. Gaweda, *University of Louisville, Department of Medicine, USA*

Rui Xu, *Missouri University of Science and Technology, USA*

Marina Sokolova, *Electronic Health Information Laboratory, CHEO Research Institute, Canada*

Jude Shavlik, *University of Wisconsin, USA*

Chin Kim On, *Universiti Malaysia Sabah, Malaysia*

Pedro J. García-Laencina, *Universidad Politécnica De Cartagena, Spain*

Emilio Soria Olivas, *University of Valencia, Spain*

Jose David Martín Guerrero, *University of Valencia, Spain*

List of Contributors

Alcañiz, M. / Universitat Politècnica de València, Spain	561
Aleixos, Nuria / Polytechnic University of Valencia, Spain	482
Arregui, Susana Fernández / Universidad Carlos III de Madrid, Spain	599
Au, Kin-Fan / The Hong Kong Polytechnic University, Hong Kong	387
Azevedo, Paulo Jorge / Universidade do Minho, Portugal	277
Bapi, Raju S. / University of Hyderabad, India	404
Basile, Teresa M. A. / Università degli Studi di Bari, Italy	348
Bella, Antonio / Universidad Politécnica de Valencia, Spain	128
Biba, Marenglen / Università degli Studi di Bari, Italy	348
Blasco, José / Instituto Valenciano de Investigaciones Agrarias (IVIA), Spain.....	482
Carr, Mahil / Institute for Development and Research in Banking Technology (IDRBT), India	499
Castejón-Limas, Manuel / Universidad de León, Spain.....	427
Celorrio, Sergio Jiménez / Universidad Carlos III de Madrid, Spain.....	599
Choi, Tsan-Ming / The Hong Kong Polytechnic University, Hong Kong	387
Das, Sanjoy / Kansas State University, USA	95, 109
Di Mauro, Nicola / Università degli Studi di Bari, Italy	348
Ding, Yi / Oklahoma State University, USA	457
Du, Jie / UMBC, USA	375
Esposito, Floriana / Università degli Studi di Bari, Italy.....	348
Fan, Guoliang /Oklahoma State University, USA.....	457
Farquad, M. A. H. / Institute for Development and Research in Banking Technology (IDRBT), India; University of Hyderabad, India.....	404
Ferilli, Stefano / Università degli Studi di Bari, Italy	348
Fernández, Carlos / Valencia Polytechnic University, Spain	440
Ferreira, Pedro Gabriel / Centre for Genomic Regulation, Spain	277
Ferri, Cèsar / Universidad Politécnica de Valencia, Spain	128
Figueiras-Vidal, Aníbal R. / Universidad Carlos III de Madrid, Spain	147
Flandrin, Patrick / Ecole Normale Supérieure de Lyon, France	223
Ganchev, Todor D. / University of Patras, Greece.....	195
García-Laencina, Pedro J. / Universidad Politécnica de Cartagena, Spain	147
Gaweda, Adam E. / University of Louisville, USA	265
Gómez-Sanchis, Juan / University of Valencia, Spain	482
González-Marcos, Ana / Universidad de La Rioja, Spain	427

Gorban, Alexander N. / University of Leicester, UK	28
Guerrero, Juan F. / University of Valencia, Spain	440, 482
Hernández-Orallo, José / Universidad Politécnica de Valencia, Spain	128
Honeine, Paul / Institut Charles Delaunay, France	223
Larrey-Ruiz, Jorge / Universidad Politécnica de Cartagena, Spain	147
Li, Dapeng / Kansas State University, USA	109
Magdalena, José R. / Valencia University, Spain	440
Martín, J. D. / Universitat de València, Spain	561
Martínez, Marcelino / Valencia University, Spain	440
Martínez-de-Pisón, Francisco J. / Universidad de La Rioja, Spain	427
Moltó, Enrique / Instituto Valenciano de Investigaciones Agrarias (IVIA), Spain	482
Monserrat, C. / Universitat Politècnica de València, Spain	561
Morales-Sánchez, Juan / Universidad Politécnica de Cartagena, Spain	147
Olier, Iván / Universidad Politécnica de Cataluña, Spain	176
On, Chin Kim / Universiti Malaysia Sabah, Malaysia	574
Ordieres-Meré, Joaquín / Universidad Politécnica de Madrid, Spain	427
Padberg, Frank / Saarland University, Germany	519
Panigrahi, Bijaya K. / Indian Institute of Technology, India	95, 109
Pattnaik, Shyam S. / National Institute of Technical Teachers' Training & Research, India	95
Plett, Eduard / Kansas State University at Salina, USA	109
Rada, Roy / UMBC, USA	375
Ramírez-Quintana, María José / Universidad Politécnica de Valencia, Spain	128
Ravi, V. / Institute for Development and Research in Banking Technology (IDRBT), India	499
Richard, Cédric / Institut Charles Delaunay, France	223
Rupérez, M. J. / Universitat Politècnica de València, Spain	561
Salah, Albert Ali / Centre for Mathematics and Computer Science (CWI), The Netherlands	539
Sancho-Gómez, José-Luis / Universidad Politécnica de Cartagena, Spain	147
Saudi, Azali / Universiti Malaysia Sabah, Malaysia	574
Serrano, Antonio J. / Valencia University, Spain	440
Shavlik, Jude / University of Wisconsin, USA	242
Sokolova, Marina / CHEO Research Institute, Canada	302, 325
Soria, Emilio / Valencia University, Spain	440
Sun, Zhan-Li / The Hong Kong Polytechnic University, Hong Kong	387
Szpakowicz, Stan / University of Ottawa, Canada and Polish Academy of Sciences, Poland	302, 325
Teo, Jason / Universiti Malaysia Sabah, Malaysia	574
Torrey, Lisa / University of Wisconsin, USA	242
Turbides, Tomás de la Rosa / Universidad Carlos III de Madrid, Spain	599
Vellido, Alfredo / Universidad Politécnica de Cataluña, Spain	176
Verdú-Monedero, Rafael / Universidad Politécnica de Cartagena, Spain	147
Vinaykumar, K. / Institute for Development and Research in Banking Technology (IDRBT), India	499
Wunsch, Donald C. / Missouri University of Science and Technology, USA	1
Xu, Lei / Chinese University of Hong Kong and Beijing University, PR China	60

Xu, Rui / Missouri University of Science and Technology, USA	1
Yu, Yong / The Hong Kong Polytechnic University, Hong Kong.....	387
Zinovyev, Andrei Y. / Institut Curie, France.....	28

Table of Contents

Foreword	xxiv
Preface	xxvi
 Volume I	
Chapter 1	
Exploring the Unknown Nature of Data: Cluster Analysis and Applications.....	1
<i>Rui Xu, Missouri University of Science and Technology, USA</i>	
<i>Donald C. Wunsch II, Missouri University of Science and Technology, USA</i>	
Chapter 2	
Principal Graphs and Manifolds	28
<i>Alexander N. Gorban, University of Leicester, UK</i>	
<i>Andrei Y. Zinovyev, Institut Curie, France</i>	
Chapter 3	
Learning Algorithms for RBF Functions and Subspace Based Functions.....	60
<i>Lei Xu, Chinese University of Hong Kong and Beijing University, PR China</i>	
Chapter 4	
Nature Inspired Methods for Multi-Objective Optimization	95
<i>Sanjoy Das, Kansas State University, USA</i>	
<i>Bijaya K. Panigrahi, Indian Institute of Technology, India</i>	
<i>Shyam S. Pattnaik, National Institute of Technical Teachers' Training & Research, India</i>	
Chapter 5	
Artificial Immune Systems for Anomaly Detection.....	109
<i>Eduard Plett, Kansas State University at Salina, USA</i>	
<i>Sanjoy Das, Kansas State University, USA</i>	
<i>Dapeng Li, Kansas State University, USA</i>	
<i>Bijaya K. Panigrahi, Indian Institute of Technology, India</i>	

Chapter 6	
Calibration of Machine Learning Models	128
<i>Antonio Bella, Universidad Politécnica de Valencia, Spain</i>	
<i>Cèsar Ferri, Universidad Politécnica de Valencia, Spain</i>	
<i>José Hernández-Orallo, Universidad Politécnica de Valencia, Spain</i>	
<i>María José Ramírez-Quintana, Universidad Politécnica de Valencia, Spain</i>	
Chapter 7	
Classification with Incomplete Data	147
<i>Pedro J. García-Laencina, Universidad Politécnica de Cartagena, Spain</i>	
<i>Juan Morales-Sánchez, Universidad Politécnica de Cartagena, Spain</i>	
<i>Rafael Verdú-Monedero, Universidad Politécnica de Cartagena, Spain</i>	
<i>Jorge Larrey-Ruiz, Universidad Politécnica de Cartagena, Spain</i>	
<i>José-Luis Sancho-Gómez, Universidad Politécnica de Cartagena, Spain</i>	
<i>Aníbal R. Figueiras-Vidal, Universidad Carlos III de Madrid, Spain</i>	
Chapter 8	
Clustering and Visualization of Multivariate Time Series	176
<i>Alfredo Vellido, Universidad Politécnica de Cataluña, Spain</i>	
<i>Iván Olier, Universidad Politécnica de Cataluña, Spain</i>	
Chapter 9	
Locally Recurrent Neural Networks and Their Applications.....	195
<i>Todor D. Ganchev, University of Patras, Greece</i>	
Chapter 10	
Nonstationary Signal Analysis with Kernel Machines	223
<i>Paul Honeine, Institut Charles Delaunay, France</i>	
<i>Cédric Richard, Institut Charles Delaunay, France</i>	
<i>Patrick Flandrin, Ecole Normale Supérieure de Lyon, France</i>	
Chapter 11	
Transfer Learning.....	242
<i>Lisa Torrey, University of Wisconsin, USA</i>	
<i>Jude Shavlik, University of Wisconsin, USA</i>	
Chapter 12	
Machine Learning in Personalized Anemia Treatment.....	265
<i>Adam E. Gaweda, University of Louisville, USA</i>	
Chapter 13	
Deterministic Pattern Mining On Genetic Sequences	277
<i>Pedro Gabriel Ferreira, Centre for Genomic Regulation, Spain</i>	
<i>Paulo Jorge Azevedo, Universidade do Minho, Portugal</i>	

Volume II

Chapter 14

Machine Learning in Natural Language Processing	302
<i>Marina Sokolova, CHEO Research Institute, Canada</i>	
<i>Stan Szpakowicz, University of Ottawa, Canada and Polish Academy of Sciences, Poland</i>	

Chapter 15

Machine Learning Applications in Mega-Text Processing	325
<i>Marina Sokolova, CHEO Research Institute, Canada</i>	
<i>Stan Szpakowicz, University of Ottawa, Canada and Polish Academy of Sciences, Poland</i>	

Chapter 16

FOL Learning for Knowledge Discovery in Documents.....	348
<i>Stefano Ferilli, Università degli Studi di Bari, Italy</i>	
<i>Floriana Esposito, Università degli Studi di Bari, Italy</i>	
<i>Marenglen Biba, Università degli Studi di Bari, Italy</i>	
<i>Teresa M. A. Basile, Università degli Studi di Bari, Italy</i>	
<i>Nicola Di Mauro, Università degli Studi di Bari, Italy</i>	

Chapter 17

Machine Learning and Financial Investing.....	375
<i>Jie Du, UMBC, USA</i>	
<i>Roy Rada, UMBC, USA</i>	

Chapter 18

Applications of Evolutionary Neural Networks for Sales Forecasting of Fashionable Products	387
<i>Yong Yu, The Hong Kong Polytechnic University, Hong Kong</i>	
<i>Tsan-Ming Choi, The Hong Kong Polytechnic University, Hong Kong</i>	
<i>Kin-Fan Au, The Hong Kong Polytechnic University, Hong Kong</i>	
<i>Zhan-Li Sun, The Hong Kong Polytechnic University, Hong Kong</i>	

Chapter 19

Support Vector Machine based Hybrid Classifiers and Rule Extraction thereof: Application to Bankruptcy Prediction in Banks	404
<i>M. A. H. Farquad, Institute for Development and Research in Banking Technology (IDRBT), India and University of Hyderabad, India</i>	
<i>V. Ravi, Institute for Development and Research in Banking Technology (IDRBT), India</i>	
<i>Raju S. Bapi, University of Hyderabad, India</i>	

Chapter 20	
Data Mining Experiences in Steel Industry	427
<i>Joaquín Ordieres-Meré, Universidad Politécnica de Madrid, Spain</i>	
<i>Ana González-Marcos, Universidad de La Rioja, Spain</i>	
<i>Manuel Castejón-Limas, Universidad de León, Spain</i>	
<i>Francisco J. Martínez-de-Pisón, Universidad de La Rioja, Spain</i>	
Chapter 21	
Application of Neural Networks in Animal Science.....	440
<i>Emilio Soria, Valencia University, Spain</i>	
<i>Carlos Fernández, Valencia Polytechnic University, Spain</i>	
<i>Antonio J. Serrano, Valencia University, Spain</i>	
<i>Marcelino Martínez, Valencia University, Spain</i>	
<i>José R. Magdalena, Valencia University, Spain</i>	
<i>Juan F. Guerrero, Valencia University, Spain</i>	
Chapter 22	
Statistical Machine Learning Approaches for Sports Video Mining Using Hidden Markov Models.....	457
<i>Guoliang Fan, Oklahoma State University, USA</i>	
<i>Yi Ding, Oklahoma State University, USA</i>	
Chapter 23	
A Survey of Bayesian Techniques in Computer Vision	482
<i>José Blasco, Instituto Valenciano de Investigaciones Agrarias (IVIA), Spain</i>	
<i>Nuria Aleixos, Polytechnic University of Valencia, Spain</i>	
<i>Juan Gómez-Sanchis, University of Valencia, Spain</i>	
<i>Juan F. Guerrero, University of Valencia, Spain</i>	
<i>Enrique Moltó, Instituto Valenciano de Investigaciones Agrarias (IVIA), Spain</i>	
Chapter 24	
Software Cost Estimation using Soft Computing Approaches	499
<i>K. Vinaykumar, Institute for Development and Research in Banking Technology (IDRBT), India</i>	
<i>V. Ravi, Institute for Development and Research in Banking Technology (IDRBT), India</i>	
<i>Mahil Carr, Institute for Development and Research in Banking Technology (IDRBT), India</i>	
Chapter 25	
Counting the Hidden Defects in Software Documents	519
<i>Frank Padberg, Saarland University, Germany</i>	
Chapter 26	
Machine Learning for Biometrics	539
<i>Albert Ali Salah, Centre for Mathematics and Computer Science (CWI), The Netherlands</i>	

Chapter 27

Neural Networks for Modeling the Contact Foot-Shoe Upper 561

*M. J. Rupérez, Universitat Politècnica de València, Spain**J. D. Martín, Universitat de València, Spain**C. Monserrat, Universitat Politècnica de València, Spain**M. Alcañiz, Universitat Politècnica de València, Spain***Chapter 28**Evolutionary Multi-Objective Optimization of Autonomous Mobile Robots in
Neural-Based Cognition for Behavioural Robustness 574*Chin Kim On, Universiti Malaysia Sabah, Malaysia**Jason Teo, Universiti Malaysia Sabah, Malaysia**Azali Saudi, Universiti Malaysia Sabah, Malaysia***Chapter 29**

Improving Automated Planning with Machine Learning 599

*Susana Fernández Arregui, Universidad Carlos III de Madrid, Spain**Sergio Jiménez Celorio, Universidad Carlos III de Madrid, Spain**Tomás de la Rosa Turbides, Universidad Carlos III de Madrid, Spain*

Detailed Table of Contents

Foreword	xxiv
Preface	xxvi

Volume I

Chapter 1

Exploring the Unknown Nature of Data: Cluster Analysis and Applications.....	1
<i>Rui Xu, Missouri University of Science and Technology, USA</i>	
<i>Donald C. Wunsch II, Missouri University of Science and Technology, USA</i>	

To classify objects based on their features and characteristics is one of the most important and primitive activities of human beings. The task becomes even more challenging when there is no ground truth available. Cluster analysis allows new opportunities in exploring the unknown nature of data through its aim to separate a finite data set, with little or no prior information, into a finite and discrete set of “natural,” hidden data structures. Here, the authors introduce and discuss clustering algorithms that are related to machine learning and computational intelligence, particularly those based on neural networks. Neural networks are well known for their good learning capabilities, adaptation, ease of implementation, parallelization, speed, and flexibility, and they have demonstrated many successful applications in cluster analysis. The applications of cluster analysis in real world problems are also illustrated.

Chapter 2

Principal Graphs and Manifolds	28
<i>Alexander N. Gorban, University of Leicester, UK</i>	
<i>Andrei Y. Zinovyev, Institut Curie, France</i>	

In many physical statistical, biological and other investigations it is desirable to approximate a system of points by objects of lower dimension and/or complexity. For this purpose, Karl Pearson invented principal component analysis in 1901 and found ‘lines and planes of closest fit to system of points’. The famous k-means algorithm solves the approximation problem too, but by finite sets instead of lines and planes. This chapter gives a brief practical introduction into the methods of construction of general principal objects (i.e., objects embedded in the ‘middle’ of the multidimensional data set). As a basis, the unifying framework of mean squared distance approximation of finite datasets is selected. Principal

graphs and manifolds are constructed as generalisations of principal components and k-means principal points. For this purpose, the family of expectation/maximisation algorithms with nearest generalisations is presented. Construction of principal graphs with controlled complexity is based on the graph grammar approach.

Chapter 3

Learning Algorithms for RBF Functions and Subspace Based Functions..... 60

Lei Xu, Chinese University of Hong Kong and Beijing University, PR China

Among extensive studies on radial basis function (RBF), one stream consists of those on normalized RBF (NRBF) and extensions. Within a probability theory framework, NRBF networks relates to nonparametric studies for decades in the statistics literature, and then proceeds in the machine learning studies with further advances not only to mixture-of-experts and alternatives but also to subspace based functions (SBF) and temporal extensions. These studies are linked to theoretical results adopted from studies of nonparametric statistics, and further to a general statistical learning framework called Bayesian Ying Yang harmony learning, with a unified perspective that summarizes maximum likelihood (ML) learning with the EM algorithm, RPCL learning, and BYY learning with automatic model selection, as well as their extensions for temporal modeling. This chapter outlines these advances, with a unified elaboration of their corresponding algorithms, and a discussion on possible trends.

Chapter 4

Nature Inspired Methods for Multi-Objective Optimization 95

Sanjoy Das, Kansas State University, USA

Bijaya K. Panigrahi, Indian Institute of Technology, India

Shyam S. Pattnaik, National Institute of Technical Teachers' Training & Research, India

This chapter focuses on the concepts of dominance and Pareto-optimality. It then addresses key issues in applying three basic classes of nature inspired algorithms – evolutionary algorithms, particle swarm optimization, and artificial immune systems, to multi-objective optimization problems. As case studies, the most significant multi-objective algorithm from each class is described in detail. Two of these, NSGA-II and MOPSO, are widely used in engineering optimization, while the others show excellent performances. As hybrid algorithms are becoming increasingly popular in optimization, this chapter includes a brief discussion of hybridization within a multi-objective framework.

Chapter 5

Artificial Immune Systems for Anomaly Detection..... 109

Eduard Plett, Kansas State University at Salina, USA

Sanjoy Das, Kansas State University, USA

Dapeng Li, Kansas State University, USA

Bijaya K. Panigrahi, Indian Institute of Technology, India

This chapter introduces anomaly detection algorithms analogous to methods employed by the vertebrate immune system, with an emphasis on engineering applications. The basic negative selection approach, as well as its major extensions, is introduced. The chapter next proposes a novel scheme to classify all

algorithmic extensions of negative selection into three basic classes: self-organization, evolution, and proliferation. In order to illustrate the effectiveness of negative selection based algorithms, one recent algorithm, the proliferating V-detectors method, is taken up for further study. It is applied to a real world anomaly detection problem in engineering, that of automatic testing of bearing machines. As anomaly detection can be considered as a binary classification problem, in order to further show the usefulness of negative selection, this algorithm is then modified to address a four-category problem, namely the classification of power signals based on the type of disturbance.

Chapter 6

Calibration of Machine Learning Models	128
--	-----

Antonio Bella, Universidad Politécnica de Valencia, Spain

Cèsar Ferri, Universidad Politécnica de Valencia, Spain

José Hernández-Orallo, Universidad Politécnica de Valencia, Spain

María José Ramírez-Quintana, Universidad Politécnica de Valencia, Spain

The evaluation of machine learning models is a crucial step before their application because it is essential to assess how well a model will behave for every single case. In many real applications, not only is it important to know the “total” or the “average” error of the model, it is also important to know how this error is distributed and how well confidence or probability estimations are made. Many current machine learning techniques are good in overall results but have a bad distribution assessment of the error. For these cases, calibration techniques have been developed as postprocessing techniques in order to improve the probability estimation or the error distribution of an existing model. This chapter presents the most common calibration techniques and calibration measures. Both classification and regression are covered, and a taxonomy of calibration techniques is established. Special attention is given to probabilistic classifier calibration.

Chapter 7

Classification with Incomplete Data	147
---	-----

Pedro J. García-Laencina, Universidad Politécnica de Cartagena, Spain

Juan Morales-Sánchez, Universidad Politécnica de Cartagena, Spain

Rafael Verdú-Monedero, Universidad Politécnica de Cartagena, Spain

Jorge Larrey-Ruiz, Universidad Politécnica de Cartagena, Spain

José-Luis Sancho-Gómez, Universidad Politécnica de Cartagena, Spain

Aníbal R. Figueiras-Vidal, Universidad Carlos III de Madrid, Spain

Many real-word classification scenarios suffer a common drawback: missing, or incomplete, data. The ability of missing data handling has become a fundamental requirement for pattern classification because the absence of certain values for relevant data attributes can seriously affect the accuracy of classification results. This chapter focuses on incomplete pattern classification. The research works on this topic currently grows wider and it is well known how useful and efficient are most of the solutions based on machine learning. This chapter analyzes the most popular and proper missing data techniques based on machine learning for solving pattern classification tasks, trying to highlight their advantages and disadvantages.

Chapter 8

Clustering and Visualization of Multivariate Time Series 176

Alfredo Vellido, Universidad Politécnica de Cataluña, Spain

Iván Olier, Universidad Politécnica de Cataluña, Spain

The exploratory investigation of multivariate time series (MTS) may become extremely difficult, if not impossible, for high dimensional datasets. Paradoxically, to date, little research has been conducted on the exploration of MTS through unsupervised clustering and visualization. In this chapter, the authors describe generative topographic mapping through time (GTM-TT), a model with foundations in probability theory that performs such tasks. The standard version of this model has several limitations that limit its applicability. Here, the authors reformulate it within a Bayesian approach using variational techniques. The resulting variational Bayesian GTM-TT, described in some detail, is shown to behave very robustly in the presence of noise in the MTS, helping to avert the problem of data overfitting.

Chapter 9

Locally Recurrent Neural Networks and Their Applications 195

Todor D. Ganchev, University of Patras, Greece

In this chapter the author reviews various computational models of locally recurrent neurons and deliberates the architecture of some archetypal locally recurrent neural networks (LRNNs) that are based on them. Generalizations of these structures are discussed as well. Furthermore, the author points at a number of real-world applications of LRNNs that have been reported in past and recent publications. These applications involve classification or prediction of temporal sequences, discovering and modeling of spatial and temporal correlations, process identification and control, etc. Validation experiments reported in these developments provide evidence that locally recurrent architectures are capable of identifying and exploiting temporal and spatial correlations (i.e., the context in which events occur), which is the main reason for their advantageous performance when compared with the one of their non-recurrent counterparts or other reasonable machine learning techniques.

Chapter 10

Nonstationary Signal Analysis with Kernel Machines 223

Paul Honeine, Institut Charles Delaunay, France

Cédric Richard, Institut Charles Delaunay, France

Patrick Flandrin, Ecole Normale Supérieure de Lyon, France

This chapter introduces machine learning for nonstationary signal analysis and classification. It argues that machine learning based on the theory of reproducing kernels can be extended to nonstationary signal analysis and classification. The authors show that some specific reproducing kernels allow pattern recognition algorithm to operate in the time-frequency domain. Furthermore, the authors study the selection of the reproducing kernel for a nonstationary signal classification problem. For this purpose, the kernel-target alignment as a selection criterion is investigated, yielding the optimal time-frequency representation for a given classification problem. These links offer new perspectives in the field of nonstationary signal analysis, which can benefit from recent developments of statistical learning theory and pattern recognition.

Chapter 11

Transfer Learning 242

Lisa Torrey, University of Wisconsin, USA

Jude Shavlik, University of Wisconsin, USA

Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned. While most machine learning algorithms are designed to address single tasks, the development of algorithms that facilitate transfer learning is a topic of ongoing interest in the machine-learning community. This chapter provides an introduction to the goals, settings, and challenges of transfer learning. It surveys current research in this area, giving an overview of the state of the art and outlining the open problems. The survey covers transfer in both inductive learning and reinforcement learning, and discusses the issues of negative transfer and task mapping.

Chapter 12

Machine Learning in Personalized Anemia Treatment 265

Adam E. Gaweda, University of Louisville, USA

This chapter presents application of reinforcement learning to drug dosing personalization in treatment of chronic conditions. Reinforcement learning is a machine learning paradigm that mimics the trial-and-error skill acquisition typical for humans and animals. In treatment of chronic illnesses, finding the optimal dose amount for an individual is also a process that is usually based on trial-and-error. In this chapter, the author focuses on the challenge of personalized anemia treatment with recombinant human erythropoietin. The author demonstrates the application of a standard reinforcement learning method, called Q-learning, to guide the physician in selecting the optimal erythropoietin dose. The author further addresses the issue of random exploration in Q-learning from the drug dosing perspective and proposes a “smart” exploration method. Finally, the author performs computer simulations to compare the outcomes from reinforcement learning-based anemia treatment to those achieved by a standard dosing protocol used at a dialysis unit.

Chapter 13

Deterministic Pattern Mining On Genetic Sequences 277

Pedro Gabriel Ferreira, Centre for Genomic Regulation, Spain

Paulo Jorge Azevedo, Universidade do Minho, Portugal

The recent increase in the number of complete genetic sequences freely available through specialized Internet databases presents big challenges for the research community. One such challenge is the efficient and effective search of sequence patterns, also known as motifs, among a set of related genetic sequences. Such patterns describe regions that may provide important insights about the structural and functional role of DNA and proteins. Two main classes can be considered: probabilistic patterns represent a model that simulates the sequences or part of the sequences under consideration and deterministic patterns that either match or not the input sequences. In this chapter a general overview of deterministic sequence mining over sets of genetic sequences is proposed. The authors formulate an architecture that divides the mining process workflow into a set of blocks. Each of these blocks is discussed individually.

Volume II

Chapter 14

Machine Learning in Natural Language Processing 302

Marina Sokolova, CHEO Research Institute, Canada

Stan Szpakowicz, University of Ottawa, Canada and Polish Academy of Sciences, Poland

This chapter presents applications of machine learning techniques to traditional problems in natural language processing, including part-of-speech tagging, entity recognition and word-sense disambiguation. People usually solve such problems without difficulty or at least do a very good job. Linguistics may suggest labour-intensive ways of manually constructing rule-based systems. It is, however, the easy availability of large collections of texts that has made machine learning a method of choice for processing volumes of data well above the human capacity. One of the main purposes of text processing is all manner of information extraction and knowledge extraction from such large text. Machine learning methods discussed in this chapter have stimulated wide-ranging research in natural language processing and helped build applications with serious deployment potential.

Chapter 15

Machine Learning Applications in Mega-Text Processing 325

Marina Sokolova, CHEO Research Institute, Canada

Stan Szpakowicz, University of Ottawa, Canada and Polish Academy of Sciences, Poland

This chapter presents applications of machine learning techniques to problems in natural language processing that require work with very large amounts of text. Such problems came into focus after the Internet and other computer-based environments acquired the status of the prime medium for text delivery and exchange. In all cases which the authors discuss, an algorithm has ensured a meaningful result, be it the knowledge of consumer opinions, the protection of personal information or the selection of news reports. The chapter covers elements of opinion mining, news monitoring and privacy protection, and, in parallel, discusses text representation, feature selection, and word category and text classification problems. The applications presented here combine scientific interest and significant economic potential.

Chapter 16

FOL Learning for Knowledge Discovery in Documents 348

Stefano Ferilli, Università degli Studi di Bari, Italy

Floriana Esposito, Università degli Studi di Bari, Italy

Marenglen Biba, Università degli Studi di Bari, Italy

Teresa M. A. Basile, Università degli Studi di Bari, Italy

Nicola Di Mauro, Università degli Studi di Bari, Italy

This chapter proposes the application of machine learning techniques, based on first-order logic as a representation language, to the real-world application domain of document processing. First, the tasks and problems involved in document processing are presented, along with the prototypical system DOMINUS and its architecture, whose components are aimed at facing these issues. Then, a closer look is provided for the learning component of the system, and the two sub-systems that are in charge of performing su-

pervised and unsupervised learning as a support to the system performance. Finally, some experiments are reported that assess the quality of the learning performance. This is intended to prove to researchers and practitioners of the field that first-order logic learning can be a viable solution to tackle the domain complexity, and to solve problems such as incremental evolution of the document repository.

Chapter 17

Machine Learning and Financial Investing..... 375

Jie Du, UMBC, USA

Roy Rada, UMBC, USA

This chapter presents the case for knowledge-based machine learning in financial investing. Machine learning here, while it will exploit knowledge, will also rely heavily on the evolutionary computation paradigm of learning, namely reproduction with change and selection of the fit. The chapter will begin with a model for financial investing and then review what has been reported in the literature as regards knowledge-based and machine-learning-based methods for financial investing. Finally, a design of a financial investing system is described which incorporates the key features identified through the literature review. The emerging trend of incorporating knowledge-based methods into evolutionary methods for financial investing suggests opportunities for future researchers.

Chapter 18

Applications of Evolutionary Neural Networks for Sales Forecasting of Fashionable Products 387

Yong Yu, The Hong Kong Polytechnic University, Hong Kong

Tsan-Ming Choi, The Hong Kong Polytechnic University, Hong Kong

Kin-Fan Au, The Hong Kong Polytechnic University, Hong Kong

Zhan-Li Sun, The Hong Kong Polytechnic University, Hong Kong

The evolutionary neural network (ENN), which is the hybrid combination of evolutionary computation and neural network, is a suitable candidate for topology design, and is widely adopted. An ENN approach with a direct binary representation to every single neural network connection is proposed in this chapter for sales forecasting of fashionable products. In this chapter, the authors will first explore the details on how an evolutionary computation approach can be applied in searching for a desirable network structure for establishing the appropriate sales forecasting system. The optimized ENN structure for sales forecasting is then developed. With the use of real sales data, the authors compare the performances of the proposed ENN forecasting scheme with several traditional methods which include artificial neural network (ANN) and SARIMA. The authors obtain the conditions in which their proposed ENN outperforms other methods. Insights regarding the applications of ENN for forecasting sales of fashionable products are generated. Future research directions are outlined.

Chapter 19

Support Vector Machine based Hybrid Classifiers and Rule Extraction thereof:

Application to Bankruptcy Prediction in Banks 404

M. A. H. Farquad, Institute for Development and Research in Banking Technology (IDRBT),

India and University of Hyderabad, India

V. Ravi, Institute for Development and Research in Banking Technology (IDRBT), India

Raju S. Bapi, University of Hyderabad, India

Support vector machines (SVMs) have proved to be a good alternative compared to other machine learning techniques specifically for classification problems. However just like artificial neural networks (ANN), SVMs are also black box in nature because of its inability to explain the knowledge learnt in the process of training, which is very crucial in some applications like medical diagnosis, security and bankruptcy prediction etc. In this chapter a novel hybrid approach for fuzzy rule extraction based on SVM is proposed. This approach handles rule-extraction as a learning task, which proceeds in two major steps. In the first step the authors use labeled training patterns to build an SVM model, which in turn yields the support vectors. In the second step extracted support vectors are used as input patterns to fuzzy rule based systems (FRBS) to generate fuzzy “if-then” rules. To study the effectiveness and validity of the extracted fuzzy rules, the hybrid SVM+FRBS is compared with other classification techniques like decision tree (DT), radial basis function network (RBF) and adaptive network based fuzzy inference system. To illustrate the effectiveness of the hybrid developed, the authors applied it to solve a bank bankruptcy prediction problem. The dataset used pertain to Spanish, Turkish and US banks. The quality of the extracted fuzzy rules is evaluated in terms of fidelity, coverage and comprehensibility.

Chapter 20

Data Mining Experiences in Steel Industry 427

Joaquín Ordieres-Meré, Universidad Politécnica de Madrid, Spain

Ana González-Marcos, Universidad de La Rioja, Spain

Manuel Castejón-Limas, Universidad de León, Spain

Francisco J. Martínez-de-Pisón, Universidad de La Rioja, Spain

This chapter reports five experiences in successfully applying different data mining techniques in a hot-dip galvanizing line. Engineers working in steelmaking have traditionally built mathematical models either for their processes or products using classical techniques. Their need to continuously cut costs down while increasing productivity and product quality is now pushing the industry into using data mining techniques so as to gain deeper insights into their manufacturing processes. The authors' work was aimed at extracting hidden knowledge from massive data bases in order to improve the existing control systems. The results obtained, though small at first glance, lead to huge savings at such high volume production environment. The effective solutions provided by the use of data mining techniques along these projects encourages the authors to continue applying this data driven approach to frequent hard-to-solve problems in the steel industry.

Chapter 21

Application of Neural Networks in Animal Science..... 440

Emilio Soria, Valencia University, Spain

Carlos Fernández, Valencia Polytechnic University, Spain

Antonio J. Serrano, Valencia University, Spain

Marcelino Martínez, Valencia University, Spain

José R. Magdalena, Valencia University, Spain

Juan F. Guerrero, Valencia University, Spain

Stock breeding has been one of the most important sources of food and labour throughout human history. Every advance in this field has always led to important and beneficial impacts on human society.

These innovations have mainly taken place in machines or genetics, but data analysis has been somewhat ignored. Most of the published works in data analysis use linear models, and there are few works in the literature that use non-linear methods for data processing in stock breeding where these methods have proven to obtain better results and performance than linear, classical methods. This chapter demonstrates the use of non-linear methods by presenting two practical applications: milk yield production in goats, and analysis of farming production systems.

Chapter 22

Statistical Machine Learning Approaches for Sports Video Mining Using Hidden Markov Models	457
---	-----

Guoliang Fan, Oklahoma State University, USA

Yi Ding, Oklahoma State University, USA

This chapter summarizes the authors' recent research on the hidden Markov model (HMM)-based machine learning approaches to sports video mining. They will advocate the concept of semantic space that provides explicit semantic modeling at three levels, high-level semantics, mid-level semantic structures, and low-level visual features. Sports video mining is formulated as two related statistical inference problems. One is from low-level features to mid-level semantic structures, and the other is from mid-level semantic structures to high-level semantics. The authors assume that a sport video is composed of a series of consecutive play shots each of which contains variable-length frames and can be labelled with certain mid-level semantic structures. In this chapter, the authors present several HMM-based approaches to the first inference problem where the hidden states are directly associated with mid-level semantic structures at the shot level and observations are the visual features extracted from frames in a shot. Specifically, they will address three technical issues about HMMs: (1) how to enhance the observation model to handle variable-length frame-wise observations; (2) how to capture the interaction between multiple semantic structures in order to improve the overall mining performance; (3) how to optimize the model structure and to learn model parameters simultaneously. This work is the first step toward the authors' long-term goal that is to develop a general sports video mining framework with explicit semantic modeling and direct semantic computing.

Chapter 23

A Survey of Bayesian Techniques in Computer Vision	482
--	-----

José Blasco, Instituto Valenciano de Investigaciones Agrarias (IVIA), Spain

Nuria Aleixos, Polytechnic University of Valencia, Spain

Juan Gómez-Sanchis, University of Valencia, Spain

Juan F. Guerrero, University of Valencia, Spain

Enrique Moltó, Instituto Valenciano de Investigaciones Agrarias (IVIA), Spain

The Bayesian approach to classification is intended to solve questions concerning how to assign a class to an observed pattern using probability estimations. Red, green and blue (RGB) or hue, saturation and lightness (HSL) values of pixels in digital colour images can be considered as feature vectors to be classified, thus leading to Bayesian colour image segmentation. Bayesian classifiers are also used to sort objects but, in this case, reduction of the dimensionality of the feature vector is often required prior to the analysis. This chapter shows some applications of Bayesian learning techniques in computer vision

in the agriculture and agri-food sectors. Inspection and classification of fruit and vegetables, robotics, insect identification and process automation are some of the examples shown. Problems related with the natural variability of colour, sizes and shapes of biological products, and natural illuminants are also discussed. Moreover, implementations that lead to real-time implementation are explained.

Chapter 24

Software Cost Estimation using Soft Computing Approaches 499

K. Vinaykumar, Institute for Development and Research in Banking Technology (IDRBT), India

V. Ravi, Institute for Development and Research in Banking Technology (IDRBT), India

Mahil Carr, Institute for Development and Research in Banking Technology (IDRBT), India

Software development has become an essential investment for many organizations. Software engineering practitioners have become more and more concerned about accurately predicting the cost of software products to be developed. Accurate estimates are desired but no model has proved to be successful at effectively and consistently predicting software development cost. This chapter investigates the use of the soft computing approaches in predicting the software development effort. Various statistical and intelligent techniques are employed to estimate software development effort. Further, based on the above-mentioned techniques, ensemble models are developed to forecast software development effort. Two types of ensemble models viz., linear (average) and nonlinear are designed and tested on COCOMO'81 dataset. Based on the experiments performed on the COCOMO'81 data, it was observed that the nonlinear ensemble using radial basis function network as arbitrator outperformed all the other ensembles and also the constituent statistical and intelligent techniques. The authors conclude that using soft computing models they can accurately estimate software development effort.

Chapter 25

Counting the Hidden Defects in Software Documents 519

Frank Padberg, Saarland University, Germany

The author uses neural networks to estimate how many defects are hidden in a software document. Input for the models are metrics that get collected when effecting a standard quality assurance technique on the document, a software inspection. For inspections, the empirical data sets typically are small. The author identifies two key ingredients for a successful application of neural networks to small data sets: adapting the size, complexity, and input dimension of the networks to the amount of information available for training; and using Bayesian techniques instead of cross-validation for determining model parameters and selecting the final model. For inspections, the machine learning approach is highly successful and outperforms the previously existing defect estimation methods in software engineering by a factor of 4 in accuracy on the standard benchmark. The author's approach is well applicable in other contexts that are subject to small training data sets.

Chapter 26

Machine Learning for Biometrics 539

Albert Ali Salah, Centre for Mathematics and Computer Science (CWI), The Netherlands

Biometrics aims at reliable and robust identification of humans from their personal traits, mainly for security and authentication purposes, but also for identifying and tracking the users of smarter applications.

Frequently considered modalities are fingerprint, face, iris, palmprint and voice, but there are many other possible biometrics, including gait, ear image, retina, DNA, and even behaviours. This chapter presents a survey of machine learning methods used for biometrics applications, and identifies relevant research issues. The author focuses on three areas of interest: offline methods for biometric template construction and recognition, information fusion methods for integrating multiple biometrics to obtain robust results, and methods for dealing with temporal information. By introducing exemplary and influential machine learning approaches in the context of specific biometrics applications, the author hopes to provide the reader with the means to create novel machine learning solutions to challenging biometrics problems.

Chapter 27

Neural Networks for Modeling the Contact Foot-Shoe Upper 561

M. J. Rupérez, Universitat Politècnica de València, Spain

J. D. Martín, Universitat de València, Spain

C. Monserrat, Universitat Politècnica de València, Spain

M. Alcañiz, Universitat Politècnica de València, Spain

Recently, important advances in virtual reality have made possible real improvements in computer aided design, CAD. These advances are being applied to all the fields and they have reached to the footwear design. The majority of the interaction foot-shoe simulation processes have been focused on the interaction between the foot and the sole. However, few efforts have been made in order to simulate the interaction between the shoe upper and the foot surface. To simulate this interaction, flexibility tests (characterization of the relationship between exerted force and displacement) are carried out to evaluate the materials used for the shoe upper. This chapter shows a procedure based on artificial neural networks (ANNs) to reduce the number of flexibility tests that are needed for a comfortable shoe design. Using the elastic parameters of the material as inputs to the ANN, it is possible to find a neural model that provides a unique equation for the relationship between force and displacement instead of a different characteristic curve for each material. Achieved results show the suitability of the proposed approach.

Chapter 28

Evolutionary Multi-Objective Optimization of Autonomous Mobile Robots in Neural-Based Cognition for Behavioural Robustness 574

Chin Kim On, Universiti Malaysia Sabah, Malaysia

Jason Teo, Universiti Malaysia Sabah, Malaysia

Azali Saudi, Universiti Malaysia Sabah, Malaysia

The utilization of a multi-objective approach for evolving artificial neural networks that act as the controllers for phototaxis and radio frequency (RF) localization behaviors of a virtual Khepera robot simulated in a 3D, physics-based environment is discussed in this chapter. It explains the comparison performances among the elitism without archive and elitism with archive used in the evolutionary multi-objective optimization (EMO) algorithm in an evolutionary robotics study. Furthermore, the controllers' moving performances, tracking ability and robustness also have been demonstrated and tested with four different levels of environments. The experimentation results showed the controllers allowed the robots to navigate successfully, hence demonstrating the EMO algorithm can be practically

used to automatically generate controllers for phototaxis and RF-localization behaviors, respectively. Understanding the underlying assumptions and theoretical constructs through the utilization of EMO will allow the robotics researchers to better design autonomous robot controllers that require minimal levels of human-designed elements.

Chapter 29

Improving Automated Planning with Machine Learning 599

Susana Fernández Arregui, Universidad Carlos III de Madrid, Spain

Sergio Jiménez Celorio, Universidad Carlos III de Madrid, Spain

Tomás de la Rosa Turbides, Universidad Carlos III de Madrid, Spain

This chapter reports the last machine learning techniques for the assistance of automated planning. Recent discoveries in automated planning have opened the scope of planners, from toy problems to real-world applications, making new challenges come into focus. The planning community believes that machine learning can assist to address these new challenges. The chapter collects the last machine learning techniques for assisting automated planners classified in: techniques for the improvement of the planning search processes and techniques for the automatic definition of planning action models. For each technique, the chapter provides an in-depth analysis of their domain, advantages and disadvantages. Finally, the chapter draws the outline of the new promising avenues for research in learning for planning systems.

Foreword

Most profit from innovation comes not from the initial discovery, but often from the first successful or large-scale application of the discovery. This is evident throughout history, from the first wireless telegraph to jet engines for mass air travel, from the first pc operating systems to the development of the Internet. So it is with machine learning – the richest fruits of new science are likely to be picked after translational research from the computer to the marketplace. But where is the low-hanging fruit?

Nearly twenty years ago the term machine learning was practically synonymous with artificial neural networks, even more particularly with the multi-layer perceptron. Applications such as explosives detection, risk scoring, inverse kinematics in robotics, and optical character recognition were gaining ground. Yet other successful commercial products relied on relatively novel algorithms, such reinforcement learning used by the airline marketing tactician. Refinements of these methods are still commercially used today and in some cases are still market leaders.

Just as in the evolution of computing in the last century the bottleneck for real impact moved from hardware development, which severely constrained speed and memory, onto software and algorithms, so it seems that in this century the limiting rate for innovation with machine learning lies increasingly with creativity in the application domain. In this sense, what is of most value now is a comprehensive survey of practical applications, with pointers to new algorithms and promising directions for the future – a handbook of the application of machine learning methods.

The authorship represented in this handbook draws from the state-of-the-art with depth in methodologies and a rich breadth of application areas. In their turn, the plethora of methodologies reviewed in the edited articles is extensive and is representative of the power of non-linear modelling. Amid stalwarts such as adaptive resonance theory and the self-organising map, along with k-means and fuzzy clustering, which are now as well established in large-scale practical applications as traditional statistical linear methods ever were, there are relative newcomers to add to this arsenal, in the form of particle swarms, artificial immune systems and other optimisation tools.

The handbook covers exploratory as well as predictive modelling, frequentist and Bayesian methods which form a fruitful branch of machine learning in their own right. This extends beyond the design of non-linear algorithms to encompass also their evaluation, a critical and often neglected area of research, yet a critical stage in practical applications. Another feature of the real-world is missing data, again an open question as regards flexible and semi-parametric models. Alongside data issues is the important matter of anomaly detection. This is not only because fraud and other hostile behaviour is *perforce* only recorded exceptionally and usually adapts over time to combat filters put in place to deny this type of behaviour, but also because every inference model, whether linear or not, becomes unreliable outside of its evidence base. For data based models, automatic novelty detection is a requirement, if it this is seldom formally done.

Time series, text, natural language, finance and retail, biometrics and computational medicine, software quality control and heavy industry, dairy products and biomechanics, robotics and computer vision – the

handbook makes a comprehensive map of the fast expansion in machine learning methodologies and their practical applications. A methodological trend that is clear is to move from globally linear to local linear transformations. A second trend is the increasing reliance on kernels. Both approaches move on from the mantra of generic flexible models and occupy the new high ground in machine learning, by shaping together the need for flexibility to model non-linear surfaces, with transparency and control coming from carefully constrained methodological principles.

In summary, this handbook is a clear and lucid introduction to the promising new world of machine learning applications. It will remain a valuable reference, even though it is more than likely that the many of the ideas it introduces will come to fruition much sooner than another twenty years from now!

*Paulo J.G. Lisboa
Liverpool John Moores University, UK*

Paulo J. G. Lisboa received the Ph.D. degree in theoretical physics from Liverpool University, Liverpool, U.K., in 1983. He was a Postdoctoral Fellow at Bristol University, Bristol, U.K., before joining the electricity generation industry to research into process control, which he taught at Liverpool University since 1987. In 1996, he was appointed to the Chair of Industrial Mathematics at Liverpool John Moores University. He was Head of the Graduate School and he is currently Professor in Industrial Mathematics and Head of Research for Statistics and Neural Computing. His research interests are in medical decision support for personalised medicine and computational marketing. He has led European and national projects, including the cancer track in the network of excellence BioPattern. Dr. Lisboa has over 190 refereed publications and is an Associate Editor for Neural Networks, Neural Computing Applications, Applied Soft Computing and Source Code for Biology and Medicine. He is also on the executive committees of the Healthcare Technologies Professional Network of the IEE and of the Royal Academy of Engineering's U.K. Focus for Biomedical Engineering.

Preface

Machine learning (ML) is one of the most fruitful fields of research currently, both in the proposal of new techniques and theoretic algorithms and in their application to real-life problems. From a technological point of view, the world has changed at an unexpected pace; one of the consequences is that it is possible to use high-quality and fast hardware at a relatively cheap price. The development of systems that can be adapted to the environment in a smart way has a huge practical application. Usually, these systems work by optimizing the performance of a certain algorithm/technique according to a certain maximization/minimization criterion but using experimental data rather than a given “program” (in the classical sense). This way of tackling the problem usually stems from characteristics of the task, which might be time-variant or too complex to be solved by a sequence of sequential instructions. Lately, the number of published papers, patents and practical applications related to ML has increased exponentially. One of the most attractive features of ML is that it brings together knowledge from different fields, such as, pattern recognition (neural networks, support vector machines, decision trees, reinforcement learning, ...), data mining (time series prediction, modeling, ...), statistics (Bayesian methods, Montecarlo methods, bootstrapping, ...) or signal processing (Markov models), among others. Therefore, ML takes advantage of the synergy of all these fields thus providing robust solutions that use different fields of knowledge.

Therefore, ML is a multidisciplinary topic and it needs some particular bibliography to gather its different techniques. There are a number of excellent references to go into ML, but the wide range of applications of ML has not been discussed in any reference book. Both theory and practical applications are discussed in this handbook. Different state-of-the-art techniques are analyzed in the first part of the handbook, while a wide and representative range of practical applications are shown in the second part of the book. The editors would like to thank the collaboration of the authors of the first part of the handbook, who accepted the suggestion of including a section of applications in their chapters.

A short introduction to the chapters will be provided in the following. The first part of the handbook consists of eleven chapters. In chapter 1, R. Xu and D. C. Wunsch II do a review of clustering algorithms and provide some real applications; it should be pointed out the adequacy of the references, which show up the deep knowledge of the authors in these techniques, ratified by the book published by the authors recently in 2008. In chapter 2, “Principal Graphs and Manifolds,” Gorban and Zinovyev present the machine learning approaches to the problem of dimensionality reduction with controlled complexity. They start with classical techniques, such as principal component analysis (PCA) and the k-means algorithm. There is a whole universe of approximants between the ‘most rigid’ linear manifolds (principal components) and ‘most soft’ unstructured finite sets of k-means. This chapter gives a brief practical introduction into the methods of construction of general principal objects (i.e., objects embedded in the ‘middle’ of the multidimensional data set). The notions of self-consistency and coarse-grained self-consistency give the general probabilistic framework for construction of principal objects. The family of expectation/maximization algorithms with nearest generalizations is presented. Construction of principal graphs

with controlled complexity is based on the graph grammar approach. In the theory of principal curves and manifolds the penalty functions were introduced to penalize deviation from linear manifolds. For branched principal object the pluriharmonic embeddings ('pluriharmonic graphs') serve as 'ideal objects' instead of planes, and deviation from this ideal form is penalized. Chapter 3, "Learning Algorithms for RBF Functions and Subspace Based Functions," by Lei Xu, overviewed advances on normalized radial basis function (RBF) and alternative mixture-of-experts as well as further developments to subspace based functions (SBF) and temporal extensions. These studies are linked to a general statistical learning framework that summarizes not only maximum likelihood learning featured by the EM algorithm but also Bayesian Ying Yang (BYY) harmony and rival penalized competitive learning (RPCL) featured by their automatic model selection nature, with a unified elaboration of their corresponding algorithms. Finally, remarks have also been made on possible trends.

Sanjoy Das, Bijaya K. Panigrahi and Shyam S. Patnaik show in chapter 4, "Nature Inspired Methods for Multi-Objective Optimization," an application of the three basic classes of nature inspired algorithms – evolutionary algorithms, particle swarm optimization, and artificial immune systems, to multi-objective optimization problems. As hybrid algorithms are becoming increasingly popular in optimization, this chapter also includes a brief discussion of hybridization within a multi-objective framework. In Chapter 5, "Artificial Immune Systems for Anomaly Detection," Eduard Plett, Sanjoy Das, Dapeng Li and Bijaya K. Panigrahi present anomaly detection algorithms analogous to methods employed by the vertebrate immune system, with an emphasis on engineering applications. The chapter also proposes a novel scheme to classify all algorithmic extensions of negative selection into three basic classes: self-organization, evolution, and proliferation. As anomaly detection can be considered as a binary classification problem, in order to further show the usefulness of negative selection, this algorithm is then modified to address a four-category problem, namely the classification of power signals based on the type of disturbance. Chapter 6, written by Antonio Bella, Cèsar Ferri, José Hernández-Orallo, and María José Ramírez-Quintana and entitled "Calibration of Machine Learning Models," reviews the most common calibration *techniques* and calibration *measures*. Calibration *techniques* improve the probability estimation of the model or correct its local (or global) bias, and the degree of calibration is assessed with calibration *measures*. Both classification and regression tasks are covered in this chapter, and a new taxonomy of calibration techniques is established. Chapter 7, "Classification with Incomplete Data" by Pedro J. García-Laencina, Juan Morales-Sánchez, Rafael Verdú-Monedero, Jorge Larrey-Ruiz, José-Luis Sancho-Gómez and Aníbal R. Figueiras-Vidal, deals with machine learning solutions for incomplete pattern classification; nowadays, data is generated almost everywhere: sensor networks in Mars, submarines in the deepest ocean, opinion polls about any topic, etc. Many of these real-life applications suffer a common drawback, missing or unknown data. The ability of handling missing data has become a fundamental requirement for pattern classification because inappropriate treatment of missing data may cause large errors or false results on classification. Machine learning approaches and methods imported from statistical learning theory have been most intensively studied and used in this subject. The aim of this chapter is to analyze the missing data problem in pattern classification tasks, and to summarize and compare some of the well-known methods used for handling missing values.

Chapter 8 shows that most of the existing research on multivariate time series (MTS) targets supervised prediction and forecasting problems. To date, in fact, little research has been conducted on the exploration of MTS through unsupervised clustering and visualization. Olier and Vellido describe generative topographic mapping through time (GTM-TT), a model with foundations in probability theory that performs such tasks. The standard version of this model has several limitations that limit its applicability, so, in this work, GTM-TT is reformulated within a Bayesian approach using variational techniques. The resulting variational Bayesian GTM-TT is shown to behave very robustly in the presence of noise in the

MTS, helping to avert the problem of data overfitting. Chapter 9, by Todor Ganchev entitled “Locally Recurrent Neural Networks and Their Applications,” offers a review of the various computational models of locally recurrent neurons, and surveys locally recurrent neural network (LRNN) architectures that are based on them. These locally recurrent architectures are capable of identifying and exploiting temporal and spatial correlations (i.e., the context in which events occur). This capability is the main reason for the advantageous performance of LRNN, when compared with their non-recurrent counterparts. Examples of real-world applications that rely on infinite impulse response (IIR) multilayer perceptron (MLP) neural networks, diagonal recurrent neural networks (DRNN), locally recurrent radial basis function neural networks (LRRBFNNs), locally recurrent probabilistic neural networks (LRPNNs), and that involve classification or prediction of temporal sequences, discovering and modeling of spatial and temporal correlations, process identification and control, etc., are briefly outlined. A quantitative assessment of the number of weights in a single layer of neurons, implementing different types of linkage, is presented as well. In conclusion, a brief account of the advantages and disadvantages of LRNNs is performed, and potentially promising research directions are discussed. Chapter 10, “Nonstationary signal analysis with kernel machines” by Paul Honeine, Cédric Richard and Patrick Flandrin, introduces machine learning for nonstationary signal analysis and classification. The authors show that some specific reproducing kernels allow a pattern recognition algorithm to operate in the time-frequency domain. Furthermore, the authors study the selection of the reproducing kernel for a nonstationary signal classification problem.

The last chapter of this theoretic section, chapter 11, “Transfer Learning” by Lisa Torrey and Jude Shavlik, discusses transfer learning, which is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned. While most machine learning algorithms are designed to address single tasks, the development of algorithms that facilitate transfer learning is a topic of ongoing interest in the machine-learning community. This chapter provides an introduction to the goals, formulations, and challenges of transfer learning. It surveys current research in this area, giving an overview of the state of the art and outlining the open problems. The survey covers transfer in both inductive learning and reinforcement learning, and discusses the issues of negative transfer and task mapping in depth.

The second part of the book is focused on applications of ML to real-life problems. In chapter 12, “Machine Learning in Personalized Anemia Treatment,” Adam E. Gaweda shows an interesting and ingenious application of reinforcement learning to drug dosing personalization in treatment of chronic conditions. In treatment of chronic illnesses, finding the optimal dose amount for an individual is also a process that is usually based on trial-and-error. This chapter focus on the challenge of personalized anaemia treatment with recombinant human erythropoietin and demonstrate the application of a standard reinforcement learning method, namely *Q*-learning, to guide the physician in selecting the optimal erythropoietin dose. Finally, the author shows computer simulations to compare the outcomes from reinforcement learning-based anaemia treatment to those achieved by a standard dosing protocol used at a dialysis unit.

Chapter 13, “Deterministic Pattern Mining On Genetic Sequences” by Pedro Gabriel Ferreira and Paulo Jorge Azevedo, presents an overview to the problem of mining deterministic motifs in collections of DNA or protein sequences. The large amount of available biological sequence data requires efficient techniques for motif mining. The authors start by introducing the basic concepts associated with sequence motif discovery. Next, an architecture common to discovery methods is proposed. Each of its blocks are discussed individually. Particular attention is given to different algorithmic approaches proposed in the literature. It finishes with a summary of the characteristics of the presented methods. Chapter 14, “Machine Learning in Natural Language Processing” by Marina Sokolova and Stan Szpakowicz, presents applications of ML in fundamental language-processing and linguistic problems: identify a word’s

part-of-speech, determine its meaning, find relations among parts of a sentence, and so on. Problems of that kind, once solved, are a necessary component of such higher-level language processing tasks as, for example, text summarization, question answering or machine translation. People are usually very good at such tasks; software solutions tend to be inferior. Solutions based on linguistically motivated, manually constructed rules are also quite labour-intensive. The easy availability of large collections of texts points to ML as a method of choice for building solutions based on texts data in amounts beyond the human capacity. Even the quality of those solutions, however, is no match for human performance yet.

In Chapter 15, the same authors also present applications to problems which involve the processing of very large amounts of texts. Problems best served by ML came into focus after the Internet and other computer-based environments acquired the status of the prime medium for text delivery and exchange. That is when the ability to handle extremely high volume of texts, which ML applications had not previously faced, became a major issue. The resulting set of techniques and practices, the so-called name mega-text language processing, are meant to deal with a mass of informally written, loosely edited text. The chapter makes a thorough review of the performance of ML algorithms that help solve knowledge-intensive natural language processing problems where mega-texts are a significant factor. The authors present applications which combine both scientific and economic interests. Chapter 16, “FOL Learning for Knowledge Discovery in Documents” by Stefano Ferilli, Floriana Esposito, Marenglen Biba, Teresa M.A. Basile and Nicola Di Mauro, proposes the application of machine learning techniques, based on first-order logic as a representation language, to the real-world application domain of document processing. First, the tasks and problems involved in document processing are presented, along with the prototypical system DOMINUS and its architecture, whose components are aimed at facing these issues. Some experiments are reported that assess the quality of the proposed approach. The authors like to prove to researchers and practitioners of the field that first-order logic learning can be a viable solution to tackle the domain complexity, and to solve problems such as incremental evolution of the document repository. The field of applications changes in the next chapters.

Thus, Chapter 17, “Machine Learning and Financial Investing” by Jie Du and Roy Rada, begins with a model for financial investing and then reviews the literature as regards knowledge-based and machine-learning based methods for financial investing. The claim is that knowledge bases can be incorporated into an evolutionary computation approach to financial investing to support adaptive investing, and the design of a system that does this is presented. In Chapter 18, “Applications Of Evolutionary Neural Networks For Sales Forecasting of Fahionable Products” by Yong Yu, Tsan-Ming Choi and Kin-Fan Au and Zhan-Li Sun, a theoretical framework is proposed in which the details on how an evolutionary computation approach can be applied in searching for a desirable network structure for the forecasting task are discussed. The optimized ENN structure for sales forecasting is then developed. With the use of real sales data, the performances of the proposed ENN forecasting scheme are compared with several other traditional methods which include artificial neural network and SARIMA. Insights regarding the applications of ENN for forecasting sales of fashionable products are generated.

In Chapter 19, “Support Vector Machine based Hybrid Classifiers and Rule Extraction Thereof: Application to Bankruptcy Prediction in Banks,” Farquad, Ravi and Bapi propose a hybrid rule extraction approach using support vector machine in the first phase and one of the intelligent techniques such as fuzzy rule based systems (FRBS), adaptive network based fuzzy inference system (ANFIS), decision tree (DT) and radial basis function networks (RBF) in the second phase within the framework of soft computing. They applied these hybrid classifiers to problem of bankruptcy prediction in banks. In the proposed hybrids, first phase extracts the support vectors using the training set and these support vectors are used to train FRBS, ANFIS, DT and RBF to generate rules. Empirical study is conducted using three datasets viz., Spanish banks, Turkish banks and US banks. It is concluded that the proposed hybrid

rule extraction procedure outperformed the stand-alone classifiers. Chapter 20, “Data Mining Experiences in Steel Industry” by Joaquín Ordieres-Meré, Ana González-Marcos, Manuel Castejón-Limas and Francisco J. Martínez-de-Pisón, reports five experiences in successfully applying different data mining techniques in a hot-dip galvanizing line. The work was aimed at extracting hidden knowledge from massive data bases in order to improve the existing control systems. The results obtained, though small at first glance, lead to huge savings at such high volume production environment. Fortunately, the industry has already recognized the benefits of data mining and is eager to exploit its advantages. Some editors of this handbook, together with Carlos Fernández and Juan Guerrero present the application of neural networks, specifically Multilayer Perceptrons and Self-Organizing Maps to Animal Science in Chapter 21, “Application of neural networks in Animal Science.” Two different applications are shown; first, milk yield prediction in goat herds, and second, knowledge extraction from surveys in different farms that is then used to improve the management of the farms.

Chapter 22, “Statistical Machine Learning Approaches for Sports Video Mining using Hidden Markov Models” by Guoliang Fan and Yi Ding, discusses the application of statistical machine learning approaches to sports video mining. Specifically, the authors advocate the concept of semantic space where video mining is formulated as an inference problem so that semantic computing can be accomplished in an explicit way. Several existing hidden Markov models (HMMs) are studied and compared regarding their performance. Particularly, it is proposed a new extended HMM that incorporates advantages from existing HMMs and offer more capacity, functionality and flexibility than its precedents. This chapter discusses the application of statistical machine learning approaches to sports video mining. Specifically, it is advocated the concept of semantic space where video mining is formulated as an inference problem so that semantic computing can be accomplished in an explicit way. Several existing hidden Markov models (HMMs) are studied and compared regarding their performance. Particularly, the authors propose a new extended HMM that incorporates advantages from existing HMMs and offer more capacity, functionality and flexibility than its precedents. José Blasco, Nuria Aleixos, Juan Gómez-Sanchis, Juan F. Guerrero and Enrique Moltó, in Chapter 23, “A Survey of Bayesian Techniques in Computer Vision,” show some applications of Bayesian learning techniques in computer vision. Agriculture, inspection and classification of fruit and vegetables, robotics, insect identification and process automation are some of the examples shown. Problems related to the natural variability of color, sizes and shapes of biological products, and natural illuminants are also discussed. Finally, implementations that lead to real-time implementation are explained.

Chapter 24, “Software Cost Estimation using Soft Computing Approaches” by K. Vinaykumar, V. Ravi and Mahil Carr, shows a different application. Predicting the cost of software products to be developed is the main objective of the software engineering practitioners. No model has proved to be effective, efficient and consistent in predicting the software development cost. In this chapter, the authors investigated the use of soft computing approaches for estimating software development effort. Further using intelligent techniques, linear ensembles and non-linear ensembles are developed within the framework of soft computing. The developed ensembles are tested on COCOMO’81 data. It is clear from empirical results that non-linear ensemble using radial basis function network as an arbitrator outperformed all other ensembles. Chapter 25 shows an application related to that described in the previous chapter. “Counting the Hidden Defects in Software Documents” by Frank Padberg shows the use of neural networks to estimate how many defects are hidden in a software document. Inputs for the models are metrics that get collected when effecting a standard quality assurance technique on the document, a software inspection. The author adapts the size, complexity, and input dimension of the networks to the amount of information available for training; and using Bayesian techniques instead of cross-validation for determining model parameters and selecting the final model. For inspections, the machine learning approach is highly suc-

cessful and outperforms the previously existing defect estimation methods in software engineering by a factor of 4 in accuracy on the standard benchmark. This approach is well applicable in other contexts that are subject to small training data sets. Chapter 26, “Machine Learning for Biometrics” by Albert Ali Salah, deals with an application within the field of biometrics. The recently growing field of biometrics involves matching human biometric signals in a fast and reliable manner for identifying individuals. Depending on the purpose of the particular biometric application, security or user convenience may be emphasized, resulting in a wide range of operating conditions. ML methods are heavily employed for biometric template construction and matching, classification of biometric traits with temporal dynamics, and for information fusion with the purpose of integrating multiple biometrics. The chapter on ML for biometrics reviews the hot issues in biometrics research, describes the most influential ML approaches to these issues, and identifies best practices. In particular, the chapter covers distance and similarity functions for biometric signals, subspace-based classification methods, unsupervised biometric feature learning, methods to deal with dynamic temporal signals, classifier combination and fusion for multiple biometric signals. Links to important biometric databases and code repositories are provided.

Chapter 27, “Neural Networks For Modeling The Contact Foot-Shoe Upper” by M. J. Rupérez, J. D. Martín, C. Monserrat and M. Alcañiz, shows that important advances in virtual reality make possible real improvements in footwear computer aided design. To simulate the interaction between the shoe and the foot surface, several tests are carried out to evaluate the materials used as the footwear components. This chapter shows a procedure based on artificial neural networks (ANNs) to reduce the number of tests that are needed for a comfortable shoe design. Using the ANN, it is possible to find a neural model that provides a unique equation for the characteristic curve of the materials used as shoe uppers instead of a different characteristic curve for each material. Chapter 28, “Evolutionary Multi-objective Optimization of Autonomous Mobile Robots in Neural-Based Cognition for Behavioral Robustness” by Chin Kim On, Jason Teo, and Azali Saudi, shows the utilization of a multi-objective approach for evolving artificial neural networks that act as the controllers for phototaxis and radio frequency (RF) localization behaviors of a virtual Khepera robot simulated in a 3D physics-based environment. It explains the comparison of performances between the elitism without archive and elitism with archive used in the evolutionary multi-objective optimization (EMO) algorithm in an evolutionary robotics perspective. Furthermore, the controllers’ moving performances, tracking ability and robustness have also been demonstrated and tested with four different levels of environments. The experimentation results show the controllers enable the robots to navigate successfully, hence demonstrating the EMO algorithm can be practically used to automatically generate controllers for phototaxis and RF-localization behaviors, respectively. Understanding the underlying assumptions and theoretical constructs through the utilization of EMO will allow the robotics researchers to better design autonomous robot controllers that require minimal levels of human-designed elements.

The last chapter, chapter 29, “Improving Automated Planning with Machine Learning” by Susana Fernández Arregui, Sergio Jiménez Celorio and Tomás de la Rosa Turbidez, reports the last machine learning techniques for the assistance of automated planning. Recent discoveries in automated planning have opened the scope of planners, from toy problems to real-world applications, making new challenges come into focus. The chapter collects the last machine learning techniques for assisting automated planners. For each technique, the chapter provides an in-depth analysis of their domain, advantages and disadvantages; finally, the chapter draws the outline of the new promising avenues for research in learning for planning systems.

Chapter 1

Exploring the Unknown Nature of Data: Cluster Analysis and Applications

Rui Xu

Missouri University of Science and Technology, USA

Donald C. Wunsch II

Missouri University of Science and Technology, USA

ABSTRACT

To classify objects based on their features and characteristics is one of the most important and primitive activities of human beings. The task becomes even more challenging when there is no ground truth available. Cluster analysis allows new opportunities in exploring the unknown nature of data through its aim to separate a finite data set, with little or no prior information, into a finite and discrete set of “natural,” hidden data structures. Here, the authors introduce and discuss clustering algorithms that are related to machine learning and computational intelligence, particularly those based on neural networks. Neural networks are well known for their good learning capabilities, adaptation, ease of implementation, parallelization, speed, and flexibility, and they have demonstrated many successful applications in cluster analysis. The applications of cluster analysis in real world problems are also illustrated. Portions of the chapter are taken from Xu and Wunsch (2008).

INTRODUCTION

To classify objects based on their features and characteristics is one of the most important and primitive activities of human beings. Objects displaying similar features and properties based on certain pre-specified criteria are classified into the same group or category. The properties of a specific new object can then be inferred using this classification information. For example, when we see a cat, we know immediately that it can climb trees and likes eating fish without really seeing it do so. This task becomes even more challenging when there is no ground truth available. Cluster analysis, also known as unsupervised classification or exploratory data analysis, aims to address this problem and explores

DOI: 10.4018/978-1-60566-766-9.ch001

the unknown nature of data through separating a finite data set, with little or no prior information, into a finite and discrete set of “natural,” hidden data structures (Everitt et al., 2001; Hartigan, 1975; Jain and Dubes, 1988).

Clustering focuses on the partition of data objects (patterns, entities, instances, observances, units) into a certain number of clusters (groups, subsets, or categories). However, there is no universally agreed upon and precise definition of the term cluster. Most researchers describe a cluster in terms of internal homogeneity and external separation (Everitt et al., 2001; Hansen and Jaumard, 1997; Jain and Dubes, 1988). Data objects that are in the same cluster are required to be similar to each other, while data objects belonging to different clusters should display sufficient dissimilarity. Here, we provide simple mathematical descriptions of two types of clustering, known as partitional and hierarchical clustering, based on the discussion in Hansen and Jaumard (1997).

Given a set of input patterns $\mathbf{X}=\{\mathbf{x}_1, \dots, \mathbf{x}_j, \dots, \mathbf{x}_N\}$, where $\mathbf{x}_j=(x_{j1}, x_{j2}, \dots, x_{jd}) \in \mathbb{R}^d$, with each measure x_{ji} called a feature (attribute, dimension, or variable):

1. Hard partitional clustering attempts to seek a K -partition of \mathbf{X} , $C=\{C_1, \dots, C_K\}$ ($K \leq N$), such that
 - $C_i \neq \emptyset$, $i = 1, \dots, K$ (1)
 - $\bigcup_{i=1}^K C_i = \mathbf{X}$ (2)
 - $C_i \cap C_j = \emptyset$, $i, j = 1, \dots, K$ and $i \neq j$ (3)
2. Hierarchical clustering attempts to construct a tree-like nested structure partition of \mathbf{X} , $H=\{H_1, \dots, H_Q\}$ ($Q \leq N$), such that $C_i \in H_m$, $C_j \in H_l$, and $m > l$ imply $C_i \subset C_j$ or $C_i \cap C_j = \emptyset$ for all $i, j \neq i$, $m, l = 1, \dots, Q$.

As shown in Eq. 3, each data object is only associated with a single cluster. In some cases, it may also be possible that an object is related to several clusters. This can be achieved by allowing the object to belong to all K clusters with a degree of membership, $u_{ij} \in [0, 1]$, which represents the membership coefficient of the j^{th} object in the i^{th} cluster and satisfies the following two constraints (Zadeh, 1965):

$$\sum_{i=1}^K u_{ij} = 1, \quad \forall j, \quad (4)$$

and

$$\sum_{j=1}^N u_{ij} < N, \quad \forall i, \quad (5)$$

This is known as fuzzy clustering and is especially useful when the clusters are not well separated and the boundaries are ambiguous (Bezdek, 1981).

The basic procedure of cluster analysis is summarized in the following steps:

1. *Feature selection or extraction.* Feature selection chooses distinguishing features from a set of candidates, while feature extraction utilizes some transformations to generate useful and novel features from the original ones (Jain et al., 1999; Bishop, 1995). Effective selection or generation of salient features can greatly decrease the storage requirement and measurement cost, simplify the

subsequent design process and computational complexity, and facilitate understanding of the data. Note that in the literature, these two terms sometimes are used interchangeably without further identifying the difference.

2. *Clustering algorithm design or selection.* This step usually is related to the determination of an appropriate proximity (similarity or distance) measure and construction of a criterion function. Intuitively, data objects are clustered based on the resemblance of each other, measured in terms of a pre-specified similarity or distance function. Once the proximity measure is determined, clustering could be further construed as an optimization problem with a specific criterion function.
3. *Cluster validation.* Given a data set, each clustering algorithm can always produce a partition whether or not there really exists a particular structure in the data. Moreover, different clustering approaches usually lead to different clusters of data, and even for the same algorithm, different parameter selection or the presentation order of input patterns may affect the final results. Therefore, effective evaluation standards and criteria are critically important to provide users with a degree of confidence for the clustering results.
4. *Result interpretation.* The ultimate goal of clustering is to provide users with meaningful insights from the original data so that they develop a clear understanding of the data and therefore effectively solve the problems encountered. Experts in the relevant fields are encouraged to interpret the data partition, integrating other experimental evidence and domain information without restricting their observations and analysis to any specific clustering result. Consequently, further analyses and experiments may be required.

It is important to point out that clustering is not a one-shot process. Rather, in many circumstances, clustering requires a series of trials and repetitions of various steps in order to obtain satisfactory results. Moreover, there are no universally effective criteria to guide the selection of features and clustering schemes. Validation criteria provide some insights into the quality of clustering solutions, but even choosing an appropriate criterion is a demanding problem. It is obvious that clustering is a subjective process in nature that precludes an absolute judgment as to the relative efficacy of all clustering techniques (Baraldi and Alpaydin, 2002; Jain et al., 1999).

As a direct demonstration of the importance of clustering, there is a vast, ever-increasing literature on cluster analysis and its applications from a wide variety of disciplines, ranging from engineering and computer sciences (computational intelligence, machine learning, pattern recognition, data mining, information retrieval, web mining, mechanical engineering, electrical engineering), life and medical sciences (genetics, biology, microbiology, paleontology, psychiatry, clinic, phylogeny, pathology), and astronomy and earth sciences (geography, geology, remote sensing), to social sciences (sociology, psychology, archeology, anthropology, education) and economics (marketing, business) (Anderberg, 1973; Everitt et al., 2001; Hartigan, 1975). Scopus ® (Scopus ® is a registered trademark of Elsevier B. V.) alone reveals over 43,900 journal and conference papers using the phrase “cluster analysis” within the article title, abstract, and key words.

The remainder of the chapter is organized as follows. We first briefly review and discuss major clustering techniques rooted in machine learning, computer science, computational intelligence, and statistics. We then focus on the clustering algorithms that are based on neural networks. Neural networks are well known for their good learning capabilities, adaptation, ease of implementation, parallelization, speed, and flexibility and have demonstrated many successful applications in cluster analysis. We further illustrate the applications of cluster analysis in some real world problems, such as document clustering

and the traveling salesman problem. We conclude the chapter by summarizing the major challenges in cluster analysis and the important trends in algorithm development.

CLUSTERING ALGORITHMS

Clustering has been applied in a variety of disciplines, with different requirements, objects, and standards. Therefore, different taxonomies of clustering algorithms have been generated. A rough but widely agreed-upon frame is to classify clustering techniques as either hierarchical clustering or partitional clustering based on the properties of the generated clusters (Everitt et al., 2001; Hansen and Jaumard, 1997; Jain et al., 1999; Jain and Dubes, 1988). Hierarchical clustering groups data with a sequence of nested partitions, while partitional clustering directly partitions data points into some pre-specified number of clusters without the hierarchical structure.

As aforementioned, hierarchical clustering (HC) algorithms organize data objects with a sequence of partitions, either from singleton clusters to a cluster including all individuals or vice versa. The former is known as agglomerative hierarchical clustering, and the latter is called divisive hierarchical clustering. The results of HC are usually depicted by a binary tree or dendrogram. The root node of the dendrogram represents the entire data set, and each leaf node is regarded as a data object. The intermediate nodes thus describe the extent that the objects are proximal to each other, and the height of the dendrogram usually expresses the distance between each pair of objects or clusters, or an object and a cluster. The distance measure is also called clustering linkage. For example, for single linkage, the distance between a pair of clusters is determined by the two closest objects to the different clusters, while the complete linkage method uses the farthest distance of a pair of objects to define inter-cluster distance. The ultimate clustering results can be obtained by cutting the dendrogram at different levels. This representation provides very informative descriptions and a visualization of the potential data clustering structures, especially when real hierarchical relations exist in the data, such as the data from evolutionary research on different species of organisms. However, classical HC algorithms lack robustness and are sensitive to noise and outliers. Moreover, the computational complexity for most HC algorithms is at least $O(N^2)$, which is unacceptable in large-scale data clustering.

In recent decades, several new HC algorithms have appeared, with the goal to improve computational complexity and robustness. One algorithm that stands out is called the BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) algorithm, which creates a new data structure, a clustering feature tree, to capture the important clustering information of the original data and to reduce the required storage (Zhang, et al., 1996). BIRCH can achieve a computational complexity of $O(N)$, which is desirable for clustering large-scale data. Another HC algorithm, known as the CURE (Clustering Using REpresentatives) algorithm (Guha et al., 1998), attempts to overcome the shape restriction of centroid-based HC by representing each cluster with a set of well-scattered points. The representative points are further shrunk towards the cluster centroid according to an adjustable parameter in order to weaken the effects of outliers. CURE utilizes a random sample (and partition) strategy to reduce computational complexity. Its counterpart in dealing with qualitative features is known as ROCK (Guha et al., 2000).

In contrast to hierarchical clustering, partitional clustering assigns a set of objects into pre-specified K clusters. One of the best-known and most popular partitional clustering algorithms is the K -means algorithm (Forgy, 1965; MacQueen, 1967). The K -means algorithm seeks an optimal partition of the data

by minimizing the sum-of-squared-error criterion with an iterative optimization procedure, which belongs to the category of hill-climbing algorithms. The basic steps of K -means are summarized as follows:

1. Initialize a K -partition randomly or based on some prior knowledge and calculate the prototype for each cluster;
2. Assign each object in the data set to the nearest cluster;
3. Recalculate the cluster prototypes based on the current partition;
4. Repeat steps 2-3 until there is no change for each cluster.

The procedure of the K -means algorithm is very simple and straightforward, and it can be easily implemented. It works very well for compact and hyperspherical clusters. The time complexity of K -means is approximately linear, which makes it a good selection for large-scale data applications. The major disadvantages of K -means come particularly from the inherent limitations when hill-climbing methods are used for optimization, such as dependence on the initial partitions, convergence problems, and sensitivity to noise and outliers. To identify the number of clusters K in advance is also a big challenge. Many variants of K -means have been proposed to address these obstacles, which can be further referenced in Xu and Wunsch (2008).

If we consider clustering from a probabilistic point of view, data points in different clusters can be assumed to be drawn from a set of underlying probability distributions. These probability sources can take different functional forms, such as multivariate Gaussian, because of its complete theory, analytical tractability, and, in many situations, natural occurrence, or they can come from the same families but with different parameters. Usually, the forms of the mixture densities are assumed to be known, which makes the process of finding the clusters of a given data set equivalent to estimating the parameters of several underlying models, where maximum likelihood (ML) estimation or Bayesian estimation can be used (Duda et al., 2001). In the case that the solutions of the likelihood equations of ML cannot be obtained analytically, the Expectation-Maximization (EM) algorithm can be utilized to approximate the ML estimates through an iterative procedure (McLachlan and Krishnan, 1997). As long as the parameters are decided, the posterior probability for assigning a data point to a cluster can be easily calculated using Bayes' theorem.

Kernel-based learning algorithms have attracted more and more efforts in recent years, especially with the introduction of support vector machines (Schölkopf et al., 1999, Vapnik, 1998). According to Cover's theorem (Cover, 1965), by nonlinearly transforming a set of complex and nonlinearly separable patterns into a higher-dimensional feature space, it is more likely to obtain a linear separation of these patterns. The difficulty of the curse of dimensionality (Bellman, 1957), which describes the exponential growth in computational complexity as a result of high dimensionality in the problem space, can be overcome by the kernel trick, arising from Mercer's theorem (Mercer, 1909). By designing and calculating an inner-product kernel, we can avoid the time-consuming, sometimes even infeasible process of explicitly describing the nonlinear mapping and computing the corresponding points in the transformed space. The commonly-used kernel functions include polynomial kernels, Gaussian radial basis function kernels, and sigmoid kernels.

Inspired by the support vector machines, Ben-Hur et al. (2001) proposed the SVC (Support Vector Clustering) algorithm in order to find a set of contours used as the cluster boundaries in the original data space. These contours can be formed by mapping back the smallest enclosing hypersphere, which contains all the data points in the transformed feature space. Chiang and Hao (2003) extended the idea

of SVC by considering each cluster as a hypersphere instead of using just one hypersphere in SVC overall. A mechanism similar to adaptive resonance theory (Carpenter and Grossberg, 1987a) is adopted to dynamically generate clusters. When an input is presented, clusters compete based on a pre-specified distance function. A validation test is performed to ensure the eligibility of the cluster to represent the input pattern. A new cluster is created as a result of the failure of all clusters available to the vigilance test. Asharaf et al. (2005) introduced the concept of a rough set into support vector clustering. In this context, the rough hypersphere has both an inner radius representing its lower approximation and an outer radius representing its upper approximation. If the mapping points in the feature space are within the lower approximation, they are considered to belong to one cluster exclusively. However, if the data points are in the upper approximation but not in the lower approximation, they are regarded to be associated with more than one cluster, which achieves the soft clustering.

NEURAL NETWORK-BASED CLUSTERING ALGORITHMS

Neural network-based clustering is closely related to the concept of competitive learning, which is traced back to the early works of Rosenblatt (1962), von der Malsburg (1973), Fukushima (1975), and Grossberg(1976a, b). According to Rumelhart and Zipser (1985, Page 76), a competitive learning scheme consists of the following three basic components:

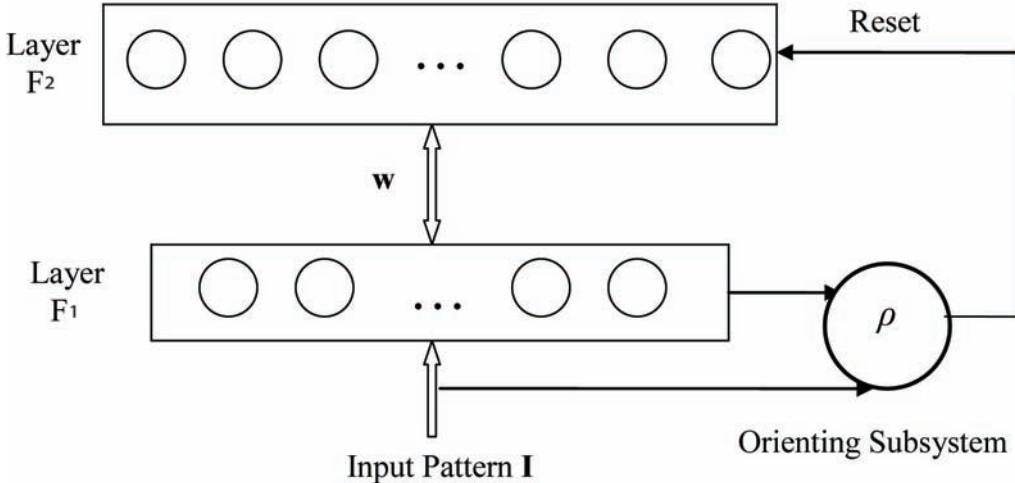
1. “Start with a set of units that are all the same except for some randomly distributed parameter which makes each of them respond slightly differently to a set of input patterns.
2. Limit the “strength” of each unit.
3. Allow the units to compete in some way for the right to respond to a given subset of inputs.”

Generally, competitive learning consists of two different categories of paradigms: winner-take-all (WTA), also known as hard or crisp competitive learning, and winner-take-most (WTM), also known as soft competitive learning (Baraldi and Blonda, 1999; Fritzke, 1997). During WTA learning, learning only occurs for a particular winning neuron that matches best with the given training input pattern. In contrast, for WTM, learning is allowed not just for the winning neuron, but all other cluster prototypes have the opportunity to be adapted based on how proximate they are to the input pattern (Baraldi and Blonda, 1999). In the following discussions, the family of adaptive resonance theory (ART) belongs to the category of WTA, while self-organizing feature maps (SOFM), neural gas, and growing neural gas adopt the WTM paradigm.

Adaptive Resonance Theory

Adaptive resonance theory (ART) was developed by Carpenter and Grossberg (1987a, 1988) as a solution to the stability-plasticity dilemma, i.e., how adaptable (plastic) should a learning system be so that it does not suffer from catastrophic forgetting of previously-learned rules (stability)? ART can learn arbitrary input patterns in a stable, fast, and self-organizing way, thus overcoming the effect of learning instability that plagues many other competitive networks. ART is not, as is popularly imagined, a neural network architecture. It is a learning theory hypothesizing that resonance in neural circuits can trigger fast learning. As such, it subsumes a large family of current and future neural network architectures with

Figure 1. Topological architecture of Fuzzy ART. Layers F_1 and F_2 are connected via adaptive weights w . The orienting subsystem is controlled by the vigilance parameter ρ .



many variants. ART1 is the first member, which only deals with binary input patterns (Carpenter and Grossberg, 1987a, 1988), although it can be extended to arbitrary input patterns by utilizing a variety of coding mechanisms. ART2 extends the application to analog input patterns (Carpenter and Grossberg, 1987b), and ART3 introduces a mechanism originating from elaborate biological processes to achieve more efficient parallel searches in hierarchical structures (Carpenter and Grossberg, 1990). Fuzzy ART (FA) incorporates fuzzy set theory and ART and can work for all real data sets (Carpenter et al., 1991). (It is typically regarded as a superior alternative to ART2.) FA generates a hyperrectangular representation of clusters. Gaussian ART (GA) adopts Gaussian-defined category choices and match functions, which monotonically increase toward the center of a cluster, to replace those of FA (Williamson, 1996). Clusters of GA take the geometric form of hyperellipsoids. Another algorithm that also uses the hyperellipsoidal representation is called ellipsoid ART (Anagnostopoulos and Georgopoulos, 2001). Baraldi and Alpaydin (2002) generalized GA in their defined constructive incremental clustering framework, called simplified ART (SART), which includes two other ART networks known as symmetric fuzzy ART (SFART) and fully self-organizing SART (FOSART) networks. It is interesting to point out that FOSART uses a “soft-to-hard competitive model transition” to minimize the distortion error (Baraldi and Alpaydin, 2002), which displaces it from the category of hard competitive learning to which other ART networks belong. Furthermore, Linares-Barranco et al. (1998) demonstrated the hardware implementations and very-large-scale integration (VLSI) design of ART systems. In Wunsch (1991) and Wunsch et al. (1993), the optical correlator-based ART implementation, instead of the implementation of ART in electronics, was also discussed. We will focus on FA in the following discussions.

As depicted in Fig. 1, the basic Fuzzy ART architecture consists of two-layer nodes or neurons, the feature representation field F_1 , and the category representation field F_2 . The neurons in layer F_1 are activated by the input pattern, while the prototypes of the formed clusters are stored in layer F_2 . The neurons in layer F_2 that are already being used as representations of input patterns are said to be committed. Correspondingly, the uncommitted neuron encodes no input patterns. The two layers are connected via

adaptive weights \mathbf{w}_j , emanating from node j in layer F_2 , which are initially set as 1. After an input pattern is presented, the neurons (including a certain number of committed neurons and one uncommitted neuron) in layer F_2 compete by calculating the category choice function (See Figure 1)

$$T_j = \frac{|\mathbf{x} \wedge \mathbf{w}_j|}{\alpha + |\mathbf{w}_j|}, \quad (6)$$

where \wedge is the fuzzy AND operator defined by

$$(\mathbf{x} \wedge \mathbf{y})_i = \min(x_i, y_i), \quad (7)$$

and $\alpha > 0$ is the choice parameter to break the tie when more than one prototype vector is a fuzzy subset of the input pattern, based on the winner-take-all rule,

$$T_J = \max_j \{T_j\}. \quad (8)$$

The winning neuron, J , then becomes activated, and an expectation is reflected in layer F_1 and compared with the input pattern. The orienting subsystem with the pre-specified vigilance parameter ρ ($0 \leq \rho \leq 1$) determines whether the expectation and the input pattern are closely matched. The larger the value of ρ , the fewer mismatches will be tolerated; therefore, more clusters are likely to be generated. If the match meets the vigilance criterion,

$$\rho \leq \frac{|\mathbf{x} \wedge \mathbf{w}_J|}{|\mathbf{x}|}, \quad (9)$$

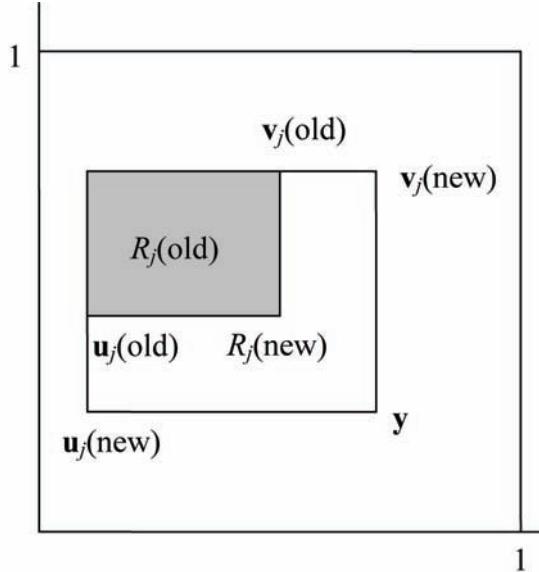
weight adaptation occurs, where learning starts and the weights are updated using the following learning rule,

$$\mathbf{w}_J(\text{new}) = \beta(\mathbf{x} \wedge \mathbf{w}_J(\text{old})) + (1 - \beta)\mathbf{w}_J(\text{old}), \quad (10)$$

where $\beta \in [0, 1]$ is the learning rate parameter and $\beta=1$ corresponds to fast learning. This procedure is called resonance, which suggests the name of ART. Carpenter et al. (1991) introduced a method, called fast-commit slow-recode, for achieving efficient coding of noisy input patterns. In this context, β is set to one when an uncommitted node is selected to represent the current input pattern. Correspondingly, Eq. 10 becomes

$$\mathbf{w}_J(\text{new}) = \mathbf{x}, \quad (11)$$

Figure 2. Category update of FA with complement coding and fast learning. Each category j is geometrically represented as a rectangle R_j . The shaded rectangle expands to the smallest rectangle to incorporate the presented input pattern \mathbf{x} into the cluster.



which indicates that the input pattern is directly copied as the prototype of the new cluster. On the other hand, committed prototypes are updated with a slow learning rate, $\beta < 1$, to prevent them from being corrupted by noise. If the vigilance criterion is not met, a reset signal is sent back to layer F_2 to shut off the current winning neuron, which will remain disabled for the entire duration of the presentation of this input pattern, and a new competition is performed among the remaining neurons. This new expectation is then projected into layer F_1 , and this process repeats until the vigilance criterion is met. In the case that an uncommitted neuron is selected for coding, a new uncommitted neuron is created to represent a potential new cluster.

A practical problem in applying FA is the possibility of cluster proliferation, which occurs as a result of an arbitrarily small norm of input patterns (Moore, 1989; Carpenter et al., 1991). Since the norm of weight vectors does not increase during learning, many low-valued prototypes may be generated without further access. The solution to the cluster proliferation problem is to normalize the input patterns (Carpenter et al., 1991) so that,

$$|\mathbf{x}| = \varsigma, \quad \varsigma > 0. \quad (12)$$

This is an extended step of the preprocessing phase.

One way to normalize an input pattern is called complement coding (Carpenter et al., 1991). Specifically, an input pattern d -dimensional $\mathbf{x} = (x_1, \dots, x_d)$ is expanded as a $2d$ -dimensional vector

$$\mathbf{x}^* = (\mathbf{x}, \mathbf{x}^c) = (x_1, \dots, x_d, x_1^c, \dots, x_d^c), \quad (13)$$

where $x_i^c = 1 - x_i$ for all i . A direct mathematical manipulation shows that input patterns in complement coding form are automatically normalized,

$$|\mathbf{x}^*| = |(\mathbf{x}, \mathbf{x}^c)| = \sum_{i=1}^d x_i + \sum_{i=1}^d x_i^c = \sum_{i=1}^d x_i + d - \sum_{i=1}^d x_i = d. \quad (14)$$

Corresponding to the expansion of the input patterns, now, the adaptive weight vectors \mathbf{w}_j are also in the $2d$ -dimensional form, represented as,

$$\mathbf{w}_j = (\mathbf{u}_j, \mathbf{v}_j^c). \quad (15)$$

Initially, \mathbf{w}_j is still set to one, which causes \mathbf{u}_j to be set to one and \mathbf{v}_j to be set to zero. The adaptation of \mathbf{w}_j also follows the same rule in Eq. 10.

A 2-dimensional geometric interpretation of FA cluster update with complement coding and fast learning is illustrated in Fig. 2, where each category is represented as a rectangle. Note that complement coding has the advantage that one can inspect templates to determine how much learning has occurred. This, together with the normalization property, has popularized complement coding in ART1 as well.

As can be seen in Fig. 2, \mathbf{u}_j and \mathbf{v}_j in Eq. 15 are both 2-dimensional vectors defining two corners of rectangle R_j , which is considered a geometric representation of cluster j . The size of R_j can be calculated using

$$|R_j| = |\mathbf{v}_j - \mathbf{u}_j|. \quad (16)$$

Note that when an uncommitted node j is eligible to encode an input pattern $\mathbf{x}^* = (\mathbf{x}, \mathbf{x}^c)$, the fast learning in Eq. 11 leads to

$$\mathbf{w}_j(\text{new}) = \mathbf{x}^* = (\mathbf{x}, \mathbf{x}^c), \quad (17)$$

which implies that both \mathbf{u}_j and \mathbf{v}_j are equal to \mathbf{x} . In this situation, rectangle R_j coincides with the point \mathbf{x} with zero size. (See Figure 2)

Suppose cluster j corresponding to the shaded area is now eligible to encode a new input pattern $\mathbf{y}^* = (\mathbf{y}, \mathbf{y}^c)$. Again, following the fast learning rule in Eq. 11, we have

$$\begin{aligned} \mathbf{w}_j(\text{new}) &= \mathbf{y}^* \wedge \mathbf{w}_j(\text{old}) \\ &= (\mathbf{y} \wedge \mathbf{u}_j(\text{old}), \mathbf{y}^c \wedge \mathbf{v}_j^c(\text{old})) \\ &= \left(\mathbf{y} \wedge \mathbf{u}_j(\text{old}), (\mathbf{y} \vee \mathbf{v}_j(\text{old}))^c \right) \\ &= (\mathbf{u}_j(\text{new}), \mathbf{v}_j^c(\text{new})) \end{aligned}, \quad (18)$$

where \vee represents the fuzzy OR operator

$$(\mathbf{x} \vee \mathbf{y})_i = \max(x_i, y_i). \quad (19)$$

As can be seen from Eq. 18, the rectangle R_j expands with the smallest size to include both the previous representation region and the new input pattern. It is also interesting to see that if \mathbf{y} is already inside R_j , there will be no change for the weight vector and, correspondingly, for the rectangle R_j .

FA exhibits many desirable characteristics, such as fast and stable learning, a transparent learning paradigm, scalability to large-scale and high-dimensional data, atypical pattern detection, and easy implementation. Huang et al. (1995) investigated and discussed more properties of FA in terms of prototype, access, reset, and the number of learning epochs required for weight stabilization.

ART may be easily used for hierarchical clustering, as well, as discussed by Wunsch (1991) and Wunsch et al. (1993). The method, called ART tree, is a hierarchy in which the same input pattern is sent to every level. Which ART units in a given level get to look at the input are determined by the winning nodes of layer F_2 at a lower level. Thus, all nodes of layer F_2 in the entire hierarchy see the same input pattern. This allows ART to perform hierarchical clustering in that the lower-level clusters will form perfect subsets of the higher-level clusters. Each layer of the ART tree has an increasing vigilance threshold so that the clusters are more finely partition as classifications become more detailed. Note, however, that due to the order dependence of inputs, and the fact that higher-level units do not all see the same set of patterns, there is a possibility of interesting classifications being achieved even if vigilance is not adjusted in this manner. The study of vigilance selection in an ART Tree hierarchy still has some worthwhile unanswered questions as of this writing. Also, two ART-based approaches for hierarchical clustering were presented by Bartfai and White (1997), known as hierarchical ART with joining (HART-J) and hierarchical ART with splitting (HART-S).

Self-Organizing Feature Maps

Self-organizing feature maps (SOFMs) developed from the work of von der Malsburg (1973), Grossberg (1976 a, b, 1978), and Kohonen (1989, 1990). Basically, the objective of SOFM is to represent high-dimensional input patterns with prototype vectors that can be visualized in, usually, a two-dimensional lattice structure, or sometimes a one-dimensional linear structure, while preserving the proximity relationships of the original data as much as possible (Kohonen, 1990, 2001). Each unit in the lattice is called a neuron, and the input patterns are fully connected to all neurons via adaptable weights. During training, neighboring input patterns are projected into the lattice, corresponding to adjacent neurons. These adjacent neurons are connected to each other, giving a clear topology of how the network fits into the input space. Therefore, the regions with a high probability of occurrence of sampled patterns will be represented by larger areas in the feature map (Haykin, 1999). In this sense, some authors prefer to think of SOFM as a method of displaying latent data structures in a visual way rather than through a clustering approach (Pal et al., 1993).

After the competition among the neurons is complete, SOFM updates a set of weight vectors within the neighborhood of the winning neuron. Learning will not occur for the neurons lying outside the neighborhood. The neighborhood is determined in a topological sense, and the size of the neighborhood is designed to decrease monotonically with time (Kohonen, 2001). Given a winning neuron J upon the presentation of an input pattern \mathbf{x} , its updating neighborhood Ω_J starts with a wide field and gradually shrinks with time until there are no other neurons inside, i.e., $\Omega_J = \emptyset$. Correspondingly, the

learning paradigm transits from soft competitive learning, which updates a neighborhood of neurons, to hard competitive learning, which only updates the winner. More specifically, we can write the updating equation for a neuron j at iteration t as,

$$\mathbf{w}_j(t+1) = \begin{cases} \mathbf{w}_j(t) + \eta(t)(\mathbf{x} - \mathbf{w}_j(t)), & \text{if } j \in \Omega_j(t) \\ \mathbf{w}_j(t), & \text{if } j \notin \Omega_j(t) \end{cases}, \quad (20)$$

where $\eta(t)$ is the monotonically decreasing learning rate. Alternately, by using the neighborhood function $h_{jj}(t)$, defined as,

$$h_{jj}(t) = \begin{cases} \eta(t), & \text{if } j \in \Omega_j(t) \\ 0, & \text{if } j \notin \Omega_j(t) \end{cases}, \quad (21)$$

Eq. 20 could be rewritten as,

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + h_{jj}(t)(\mathbf{x} - \mathbf{w}_j(t)). \quad (22)$$

The definition of $h_{jj}(t)$ can be generalized to take the form of a Gaussian function, written as,

$$h_{jj}(t) = \eta(t) \exp\left(\frac{-\|\mathbf{r}_j - \mathbf{r}_j\|^2}{2\sigma^2(t)}\right), \quad (23)$$

where \mathbf{r}_j and \mathbf{r}_j represent the positions of the corresponding neurons on the lattice, and $\sigma(t)$ is the monotonically decreasing kernel width function.

While SOFM enjoys the merits of input space density approximation and independence of the order of input patterns in a batch mode (Kohonen, 2001), the requirement for determining the number of clusters in advance and the inefficiency of handling noise and outliers may limit its applications in some real-world problems (Baraldi and Blonda, 1999). As pointed out by Haykin (1999), trained SOFM may suffer from input space density misrepresentation, where areas of low pattern density may be over-represented and areas of high density under-represented. Moreover, because SOFM only aims to preserve the topology of the inverse mapping from the lattice to the input manifold, but not necessarily of the mapping from the input manifold to the network (Martinetz and Schulten, 1994), SOFM no longer provides perfect topology preservation when the dimension of the input patterns is larger than the dimension of the output network. Kohonen (2001) provided a detailed review of a great number of SOFM variants in order to improve the performance of the basic SOFM and also to broaden its applications. These methods are based on a wide variety of considerations, such as different matching criteria, optimization methods for search improvement, dynamically defined network topology, adaptive-subspace SOFM, evolutionary-learning SOFM, SOFM for sequential data, and speed-up of SOFM.

NEURAL GAS

The neural gas (NG) algorithm also belongs to the class of self-organizing neural networks and is capable of adaptively determining the updating of the neighborhood by using a neighborhood ranking of the prototype vectors within the input space, rather than a neighborhood function in the output lattice (Martinetz et al., 1993). Given an input pattern $\mathbf{x} \in \mathbb{R}^d$, the K prototype vectors $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_K\}$ of the network are sorted based on their Euclidean distance to \mathbf{x} , i.e., $(j_0, j_1, \dots, j_{K-1})$ is a sequence of indices such that \mathbf{w}_{j_0} is the prototype vector that is closest to \mathbf{x} and $\mathbf{w}_{j_{K-1}}$ is the one that is farthest from \mathbf{x} . Let $k_j(\mathbf{x}, \mathbf{W})$ denote the number k associated with a prototype vector \mathbf{w}_j , where $k=0, \dots, K-1$ is the prototype vector for which there exist k vectors \mathbf{w}_i . With

$$\|\mathbf{x} - \mathbf{w}_i\| < \|\mathbf{x} - \mathbf{w}_{j_k}\|, \quad (24)$$

we can derive the learning rule of the prototype vectors via the optimization of a global cost function (Martinetz et al., 1993), defined as,

$$J(\mathbf{W}, \lambda) = \frac{1}{\sum_{k=0}^{K-1} h_\lambda(k)} \sum_{j=1}^K \int h_\lambda(k_j(\mathbf{x}, \mathbf{W})) (\mathbf{x} - \mathbf{w}_j)^2 P(\mathbf{x}) d^d \mathbf{x}, \quad (25)$$

where $P(\mathbf{x})$ represents the probability distribution of the data points, λ is a characteristic decay constant, and $h_\lambda(k_j(\mathbf{x}, \mathbf{W}))$ is a bell-shaped curve, written as

$$h_\lambda(k_j(\mathbf{x}, \mathbf{W})) = \exp(-k_j(\mathbf{x}, \mathbf{W})/\lambda). \quad (26)$$

By using the gradient descent method on the cost function, the prototype vectors are updated as

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \eta(t) h_\lambda(k_j(\mathbf{x}, \mathbf{W})) (\mathbf{x} - \mathbf{w}_j(t)). \quad (27)$$

A batch version of NG, equal to the minimization of the cost function with the Newton method, was also proposed (Cottrell et al., 2006). Typically, both learning rate η and characteristic decay constant λ monotonically decrease with time according to a cooling scheme. The designs, together with the definition of the h function, assure a gradual decrease of the number of updating neurons and the adjusting strength. As λ approximates zero, the learning rule in Eq. 27 becomes equivalent to that of hard competitive learning.

The major process of the NG algorithm is as follows:

1. Initialize a set of prototype vectors $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_K\}$ randomly;
2. Present an input pattern \mathbf{x} to the network. Sort the index list in order from the prototype vector with the smallest Euclidean distance from \mathbf{x} to the one with the greatest distance from \mathbf{x} ;

3. Calculate the current learning rate and $h_\lambda(k_j(\mathbf{x}, \mathbf{W}))$. Adjust the prototype vectors using the learning rule in Eq. 27;
4. Repeat steps 2 and 3 until the maximum number of iterations is reached.

For NG, the number of prototype vectors still must be decided in advance, which leads to the development of a dynamic NG algorithm, called plastic NG, whose converge properties were also investigated (Ridella et al., 1998). The NG algorithm can also be combined with the competitive Hebbian rule to construct models of topology-preserving maps (Martinetz and Schulten, 1994).

GROWING NEURAL GAS

Growing neural gas (GNG) (Fritzke, 1995, 1997) combines the growth mechanism inherited from growing cell structures (GCS) (Fritzke, 1994) with the topology generation rules of competitive Hebbian learning (Martinetz and Schulten, 1994). Each prototype vector \mathbf{w}_j is connected with its topological neighborhood via a set of edges to form an induced Delaunay triangulation (Martinetz and Schulten, 1994). Upon the presentation of an input pattern \mathbf{x} , an edge between the two closest prototype vectors with respect to \mathbf{x} is created. An edge removal mechanism, called the edge aging scheme, is designed to discard the edges that are no longer valid in the subgraph due to the adaptation of the corresponding neurons. When prototype learning occurs, not only is the prototype vector of the winning neuron J_1 updated towards \mathbf{x} , but the prototypes within its topological neighborhood N_{J_1} are also adapted, although with a smaller updating strength. Different from NG, GCS, or SOFM, which require a fixed network dimensionality *a priori*, GNG is developed as a self-organizing network that can dynamically increase (usually) and remove the number of neurons in the network. A succession of new neurons is inserted into the network every λ iterations near the neuron with the maximum accumulated error. At the same time, a neuron removal rule could also be used to eliminate the neurons featuring the lowest utility for error reduction (Baraldi and Blonda, 1999). This utility measures the increase in overall distortion error caused by the removal of the neuron of interest. Moreover, the edge aging scheme also provides a way to detect and remove the neurons that are inactive over a predefined number of iterations.

The complete GNG algorithm proceeds with the following steps:

1. Initialize a set of prototype vectors (typically 2) $\mathbf{W}=\{\mathbf{w}_1, \mathbf{w}_2\}$ randomly and a connection set \mathbf{C} to empty, $\mathbf{C} \subset \mathbf{W} \times \mathbf{W}$ and $\mathbf{C} = \emptyset$;
2. Present an input pattern \mathbf{x} to the network. Choose the winning neuron, J_1 , and the second winning neuron, J_2 , according to the Euclidean distance to \mathbf{x}

$$J_1 = \arg \min_{\mathbf{w}_j \in \mathbf{W}} \|\mathbf{x} - \mathbf{w}_j\|, \quad (28)$$

$$J_2 = \arg \min_{\mathbf{w}_j \in \mathbf{W} \setminus \{\mathbf{w}_{J_1}\}} \|\mathbf{x} - \mathbf{w}_j\|; \quad (29)$$

3. Create and add a connection between J_1 and J_2 into \mathbf{C} if it does not already exist,

$$\mathbf{C} = \mathbf{C} \cup \{(J_1, J_2)\}. \quad (30)$$

Set the age of the connection between J_1 and J_2 to zero,

$$\text{age}(J_1, J_2) = 0; \quad (31)$$

4. Update the local error of the winning neuron, J_1 ,

$$E_{J_1}(t) = E_{J_1}(t-1) + \|\mathbf{x} - \mathbf{w}_{J_1}(t)\|^2; \quad (32)$$

5. Update the prototype vectors of the winning neuron, J_1 , and its direct topological neighbors with the following rules:

$$\mathbf{w}_{J_1}(t+1) = \mathbf{w}_{J_1}(t) + \eta_w (\mathbf{x} - \mathbf{w}_{J_1}(t)), \quad (33)$$

where η_w is the learning rate for the winning neuron, and,

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \eta_n (\mathbf{x} - \mathbf{w}_j(t)), \quad \forall j \in N(J_1), \quad (34)$$

where $N(J_1)$ is the set of direct topological neighbors of J_1 , and η_n is the corresponding learning rate;

6. Increase the age of all edges emanating from J_1 by one,

$$\text{age}(J_1, j) = \text{age}(J_1, j) + 1, \quad \forall j \in N(J_1); \quad (35)$$

7. Remove the edges whose age values are larger than a prespecified threshold α_{\max} . Remove also the neurons that no longer have emanating edges as a result of the previous operation;
8. If the number of iterations is an integer multiple of the prespecified parameter λ , insert a new neuron in the network,
 - a. Choose the neuron J_e with the maximum accumulated error,

$$J_e = \arg \max_{\mathbf{w}_j \in \mathbf{W}} E_j; \quad (36)$$

- b. Choose the neuron J_f in the neighborhood of J_e with the maximum accumulated error,

$$J_f = \arg \max_{j \in N(J_e)} E_j; \quad (37)$$

c. Insert a new neuron J_n into the network,

$$\mathbf{W} = \mathbf{W} \cup \left\{ \mathbf{w}_{J_n} \right\}, \quad (38)$$

with the corresponding prototype vector initialized as

$$\mathbf{w}_{J_n} = \frac{\mathbf{w}_{J_e} + \mathbf{w}_{J_f}}{2}; \quad (39)$$

d. Insert edges that connect J_n with J_e and J_f , and remove the edge between J_e and J_f ,

$$\mathbf{C} = \left(\mathbf{C} \setminus \left\{ (J_e, J_f) \right\} \right) \cup \left\{ (J_n, J_e), (J_n, J_f) \right\}; \quad (40)$$

e. Adjust the accumulated error of J_e and J_f

$$E_{J_e} = E_{J_e} - \alpha E_{J_e}; \quad (41)$$

$$E_{J_f} = E_{J_f} - \alpha E_{J_f}; \quad (42)$$

where α is a small constant. Set the initial error of J_n ,

$$E_{J_n} = \frac{E_{J_e} + E_{J_f}}{2}; \quad (43)$$

9. Decrease the accumulated error of all neurons,

$$E_j = E_j - \beta E_j, \forall \mathbf{w}_j \in \mathbf{W}; \quad (44)$$

where β is a constant.

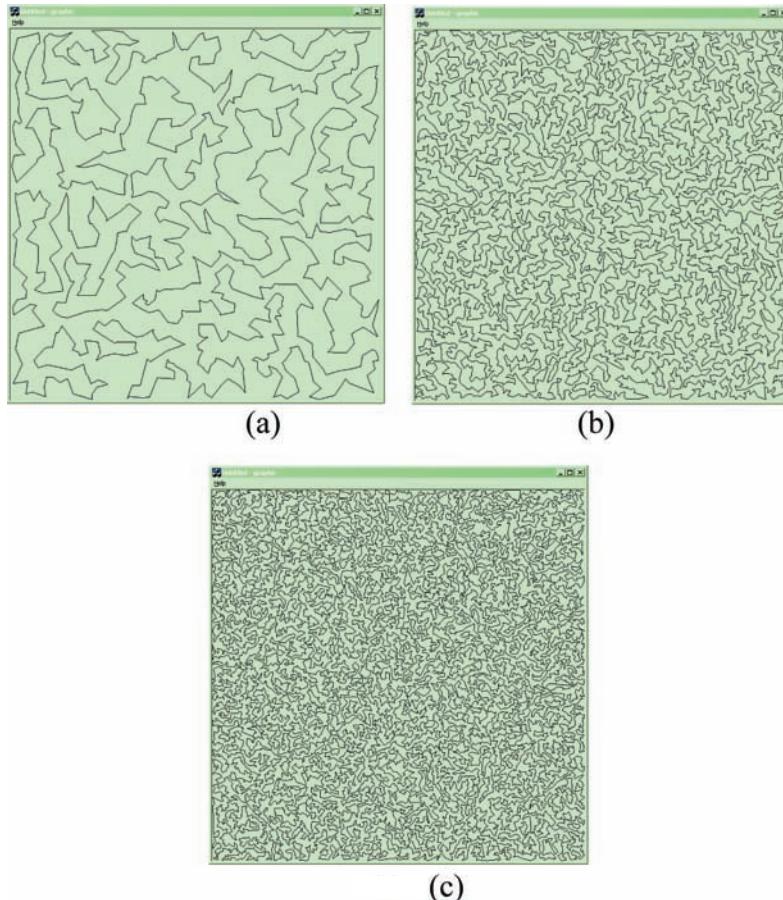
10. Repeat steps 2 through 9 until the allowed maximum network size is reached or the mean accumulated error is smaller than a prespecified threshold.

APPLICATIONS

Traveling Salesman Problem

The Traveling Salesman Problem (TSP) is one of the most studied examples in an important class of problems known as NP-complete problems. Given a complete undirected graph $G = (V,E)$, where V is a set of vertices and E is a set of edges each relating two vertices with an associated non-negative integer cost, the most general form of the TSP is equivalent to finding any Hamiltonian cycle, which is a tour over G that begins and ends at the same vertex and visits other vertices exactly once. The more common form of the problem is the optimization problem of trying to find the shortest Hamiltonian cycle, and in particular, the most common is the Euclidean version, where the vertices and edges all lie in the plane. Mulder and Wunsch (2003) applied a divide-and-conquer clustering technique, with ART networks, to scale the problem to 1 million cities, and later, to 25 million cities (Wunsch and Mulder, 2004). The divide-and-conquer paradigm provides the flexibility to hierarchically break large problems into arbitrarily small clusters depending on the desired trade-off between accuracy and speed. In addition,

Figure 3. Clustering divide-and-conquer TSP resulting tours for (a) 1k, (b) 8k, (c) 20k cities. The clustered LK algorithm achieves a significant speedup and shows good scalability.



tion, the sub-problems provide an excellent opportunity to take advantage of parallel systems for further optimization. (See Figure 3)

Specifically, the proposed algorithm combines both ART and the Lin-Kernighan (LK) local optimization algorithm (Lin and Kernighan, 1973) to divide-and-conquer instances of the TSP. As the first stage of the divide-and-conquer algorithm, an ART network is used to sort the cities into clusters, dividing the problem into smaller sub-problems. The vigilance parameter is used to set a maximum distance from the current pattern, and a vigilance parameter between 0 and 1 is considered as a percentage of the global space to determine the vigilance distance. Values are chosen based on the desired number and size of individual clusters. Each individual cluster is then passed to a version of the LK algorithm. Since the size of the tours is controlled and kept under one thousand cities, the LK is allowed to search an infinite depth as long as the total improvement for a given swap series remains positive. After a first pass through the LK algorithm, a simple intersection removal algorithm is applied. This algorithm is based on the idea that any tour containing an intersection between two edges is demonstrably sub-optimal and capable of making double-bridge swaps that the LK algorithm is unable to discover. The last step involves the combination of a number of sub-tours back into one complete tour, which is achieved by adding the tours in order of increasing distance from the origin.

Tours with good quality for city levels up to 1,000,000 were obtained within 25 minutes on a 2GHz AMD Athlon MP processor with 512M of DDR RAM. For the 25 million city problem, the algorithm takes 13,500 seconds, while Chained LK cannot solve the problem at all within the memory constraints of the machine. The visualizing results for 1,000, 8,000, and 20,000 cities are shown in Fig. 3, respectively.

It is interesting to point out that the clustering problem can also be constructed as a traveling salesman problem (McCormick et al., 1972; Lenstra, 1974). Given a d -dimensional data set with N objects, denoted as a matrix $\mathbf{X} = \{x_{ij}\}_{N \times d}$, where each row corresponds to an object and each column represents a feature of the objects, the goal of clustering is to find an optimal permutation P_0 of the rows that maximizes (minimizes) the sum of the similarities (distances) between adjacent rows. Considering the fact that in natural clusters, the intra-cluster distances are usually much less than the inter-cluster distances, it is more appropriate not to include the inter-cluster distances during the optimization computation, which leads to the following objective function (Climer and Zhang, 2006),

$$J(P_0) = \min \left(\sum_{i=1}^K \sum_{j=n_i(1)}^{n_i(L_i)-1} D(j, j+1) \right) \quad (45)$$

where K is the number of clusters, $n_i(1)$ is the first element of cluster i , $n_i(L_i)$ is the last element of cluster i , and $D(\cdot)$ is the distance function.

Assuming that each data object corresponds to a city and defining the cost between a pair of cities (objects) as the corresponding distances of the two objects, the clustering problem above is mapped as a TSP instance except that an optimal path is pursued in clustering rather than a closed tour in TSP. This problem can be resolved by introducing a dummy city that has the same distance to all the other cities (Climer and Zhang, 2006). The optimal path is achieved by breaking the derived tour at the point representing the dummy city.

Similarly, in order to realize the objective function above, K dummy cities are added, and the distances between these dummy cities are defined as infinity so that no dummy city will be connected to another dummy city in the tour (Climer and Zhang, 2006). In this way, the dummy cities serve as the

boundary of the clusters and cut the achieved tour into K segments, corresponding to K clusters. Once a TSP tour is achieved, the dummy cities, together with the adjacent edges, are removed, and the clustering boundaries are also identified.

Document Clustering

Document clustering, particularly web document clustering over the Internet, has become increasingly important as a result of the requirement for automatic creation of document hierarchy, information retrieval from document collections, and search engine result query and analysis. A comparison of the performance of agglomerative hierarchical clustering and K -means clustering (with one of its variants) on eight document data sets was performed in Steinbach et al. (2000). A phase-based incremental web document clustering system, which uses a set of sentences to describe a document rather than individual word analysis, was proposed in Hammouda and Kamel (2004).

WEBSOM is an information retrieval system based on SOFM, aiming to organize document collections and databases in a meaningful and automatic way so that users can have an easier experience browsing and exploring them (Honkela et al., 1997; Kaski et al., 1998). In this system, documents are mapped onto a two-dimensional lattice of neurons, called a document map, and those that share similar contents will be represented by the same or neighboring neurons on the map. Links are created for each neuron to provide access to the corresponding documents. Documents are first encoded as word histograms, which are then compressed via self-organizing semantic maps or word category maps to generate word clusters. The document map is formed with the reduced histograms as fingerprints of the documents. The modified version of WEBSOM, called WEBSOM2 (Kohonen et al., 2000), uses a random projection method to encode word histograms instead of word category maps. WEBSOM2 combines several speed-up methods, such as the initialization of large SOFM based on carefully constructed smaller ones and the parallelized batch map algorithm, in order to scale to large-scale document databases. In comparison with the original SOFM, the shortcut methods achieve a nine-tenths decrease of the computational time with comparable maps (Kohonen et al., 2000).

The effectiveness of WEBSOM for clustering a documental database with 6,840,568 patent abstracts was demonstrated in Kohonen et al. (2000). The number of neurons on the map is up to 1,002,240, which is the largest WEBSOM map known so far. It is interesting to point out that the generated clusters could also provide users with different aspects of the search keywords and unveil more relevant information with regard to the query. The recent advances of WEBSOM and several demonstrations of WEBSOM in news bulletins and Usenet newsgroups can be accessed at <http://websom.hut.fi/websom/>.

Investigation of the performance of ART1 on document clustering was performed in Massey (2003), where the ‘ModApte’ split of the Reuter-21578 Distribution 1.0 data set was used. A heuristic method was proposed to determine the value of the vigilance parameter, which begins with a minimal vigilance parameter and increases incrementally until a satisfactory clustering result is obtained. The minimal vigilance parameter is calculated by setting it at $1/d$, where d is the number of features. Another application of ART1 in clustering web users based on their web access patterns was illustrated in Rangarajan et al. (2004). In the system, a feature extractor first generates feature vectors from the web log files that record client requests. The feature vectors are then input into the ART1 to form client clusters, represented as the prototype vectors. When a host connects to the proxy server, all the URL objects in the cluster that the host belongs to are returned instead of a single user. Because a document may belong to more than one category, fuzzy ART was modified to adapt to this requirement and perform soft clustering (Kon-

dadadi and Kozma, 2002). In this case, after an input pattern is presented, all the nodes in the F_2 layer, rather than only the winning neuron, are checked to see whether they can pass the vigilance test. The degree of membership of the input pattern to a specific cluster is determined by the measure calculated in the vigilance test.

Neural Information Retrieval System for Group Technology

In the manufacturing industry, it is important to avoid unnecessary redesigning of parts, which can be costly in both time and money. Group technology refers to the study and implementation of information retrieval systems that can retrieve, store, and compare designs. A neural information retrieval system using ART1 networks was developed for application to this problem of group technology (Caudell et al, 1991; Caudell et al, 1994). Two or three-dimensional representations of engineering designs are input to ART1 to produce groups or families of similar parts. A new part design is then queried and compared with these families to prevent duplication of design efforts.

The generic system architecture for group technology applications consists of five basic components:

- 1) CAD system interface and parser;
- 2) Feature representation generator;
- 3) Standard orientor;
- 4) Neural network macrocircuits; and
- 5) User interaction interface.

After the design of a part has been completed, a list of instructions on how to draw and annotate a diagram of the part is stored in a file. The parser extracts the salient information from the CAD system interface, which the representation generator then converts and compresses into a form usable by ART1. A modified version of ART1 is used in the system in order to provide direct operations on compressed codes of the input vectors and the memory prototypes (Caudell et al, 1991). The orientor assures that similar parts are represented at similar locations and orientations within the graphics viewport.

The structure of ART1 macrocircuits provides an implementation of feature selection. The detailed structure of this macrocircuit evolves during training, during which a training set of part designs is repetitively presented to the networks. Within this macrocircuit, shape representation is considered first by the lowest “shape” ART1 module. For each cluster formed by this module, an additional pair of ART1 modules, known as the “holes” and “bends” modules, is spawned for secondary training. After learning has stabilized in the shape module, those parts from the training set assigned to a shape cluster are used to separately train the pair of holes and bends ART1 modules associated with the cluster. The logic at the very top of the figure intersects the names on the lists. This architecture makes it possible to choose to discriminate based on shape alone, shape and holes, shape and bends, or shape, bends, and holes.

Another user requirement is the ability to specify on-line among degrees of family similarity. This ability can be implemented with a hierarchical abstraction tree of macrocircuit modules. Each module in the tree is trained separately with the input patterns associated only with that branch cluster, and each module receives the complete set of feature representations. The modules at the top of the tree have the greatest discrimination, while the one at the bottom has the least. When a query occurs, the lowest module places the design into one of its families or clusters. Families at this level represent the most

general abstraction of the possible set of designs stored in the system. When a winning cluster is selected at the first level, the appropriate module within the next level is activated. This module places the design into one of its clusters, and the process repeats. The user selects the level of abstraction at retrieval time according to the requirements of the current design.

CONCLUSION

Cluster analysis, comprising of human exploration in exposing unknown phenomena and understanding new objects, has already attracted and will continue to attract intensive efforts from a wide variety of communities. The lack of prior information usually makes cluster analysis more difficult than supervised classification. Moreover, in essence, clustering is also a subjective process, which requires extra attention when clustering data. The assumption on the data, the definition of the proximity measure, the construction of the optimum criterion, the selection of the clustering algorithm and the corresponding parameters, and the determination of the validation index all have subjectivity. For example, given an eagle, a cardinal, a lion, a panther, and a ram, Tom is interested in finding the animals that can fly. Then the eagle and the cardinal are categorized into the same cluster, and the rest are in the other cluster. On the other hand, Sam is more interested in whether these animals are carnivores or not. In this case, the cardinal and the ram are put into the same cluster, and the other three belong to the second cluster.

As we have already seen, a broad variety of clustering algorithms has been developed, evolving from different research communities and with different goals. Each algorithm has its own pros and cons and is useful for addressing a specific category of problems. Although there are many successful applications of cluster analysis, there still remain many open problems due to the existence of many inherent, uncertain factors. It will not be surprising to see the continuous growth of clustering algorithms in the future.

We summarize and emphasize several important properties as the instruction for developing and designing new clustering algorithms, although detailed requirements for specific applications will affect these properties:

- Capability to generate arbitrary shapes of clusters rather than being confined to some particular shape.
- Scalability to large-scale data as well as high-dimensional features with reasonable time and storage complexity.
- Detection and removal of potential outliers and noise.
- Insensitivity to user-dependent parameters.
- Capability to handle newly-occurring data without re-learning from scratch.
- Immunity to the effects of the order of input patterns.
- Capability to provide insight on the number of potential clusters without prior knowledge.
- Good data visualization and clear results for further analysis.
- Capability to handle both numerical and nominal data, or easy adaptability to different data types.

REFERENCES

- Anagnostopoulos, G., & Georgopoulos, M. (2001). Ellipsoid ART and ARTMAP for incremental unsupervised and supervised learning. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'01)*, 2 (pp. 1221-1226).
- Anderberg, M. (1973). *Cluster analysis for applications*. New York: Academic Press.
- Asharaf, S., Shevade, S., & Murty, M. (2005). Rough support vector clustering. *Pattern Recognition*, 38, 1779–1783. doi:10.1016/j.patcog.2004.12.016
- Baraldi, A., & Alpaydin, E. (2002). Constructive feedforward ART clustering networks – part I and II. *IEEE Transactions on Neural Networks*, 13(3), 645–677. doi:10.1109/TNN.2002.1000130
- Baraldi, A., & Blonda, P. (1999). A survey of fuzzy clustering algorithms for pattern recognition – part I and II. *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics*, 29(6), 778–801. doi:10.1109/3477.809032
- Bartfai, G., & White, R. (1997). ART-based modular networks for incremental learning of hierarchical clusterings. *Connection Science*, 9(1), 87–112. doi:10.1080/095400997116757
- Bellman, R. (1957). *Dynamic programming*. Princeton, NJ: Princeton University Press.
- Ben-Hur, A., Horn, D., Siegelmann, H., & Vapnik, V. (2001). Support vector clustering. *Journal of Machine Learning Research*, 2, 125–137. doi:10.1162/15324430260185565
- Bezdek, J. (1981). *Pattern recognition with fuzzy objective function algorithms*. New York: Plenum Press.
- Bishop, C. (1995). *Neural networks for pattern recognition*. New York: Oxford University Press.
- Carpenter, G., & Grossberg, S. (1987a). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision Graphics and Image Processing*, 37, 54–115. doi:10.1016/S0734-189X(87)80014-2
- Carpenter, G., & Grossberg, S. (1987b). ART2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, 26(23), 4919–4930. doi:10.1364/AO.26.004919
- Carpenter, G., & Grossberg, S. (1988). The ART of adaptive pattern recognition by a self-organizing neural network. *IEEE Computer*, 21(3), 77–88.
- Carpenter, G., & Grossberg, S. (1990). ART3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks*, 3(23), 129–152. doi:10.1016/0893-6080(90)90085-Y
- Carpenter, G., Grossberg, S., & Rosen, D. (1991). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4, 759–771. doi:10.1016/0893-6080(91)90056-B

- Caudell, T., Smith, S., Escobedo, R., & Anderson, M. (1994). NIRS: Large scale ART-1 neural architectures for engineering design retrieval. *Neural Networks*, 7(9), 1339–1350. doi:10.1016/0893-6080(94)90084-1
- Caudell, T., Smith, S., Johnson, G., & Wunsch, D., II. (1991). An application of neural networks to group technology. In *Proceedings of SPIE, vol. 1469, Applications of Neural Networks II* (pp. 612-621).
- Chiang, J., & Hao, P. (2003). Anew kernel-based fuzzy clustering approach: Support vector clustering with cell growing. *IEEE transactions on Fuzzy Systems*, 11(4), 518–527. doi:10.1109/TFUZZ.2003.814839
- Climer, S., & Zhang, W. (2006). Rearrange clustering: Pitfalls, remedies, and applications. *Journal of Machine Learning Research*, 7, 919–943.
- Cottrell, M., Hammer, B., Hasenfu, A., & Villmann, T. (2006). Batch and median neural gas. *Neural Networks*, 19(6-7), 762–771. doi:10.1016/j.neunet.2006.05.018
- Cover, T. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, EC-14, 326–334. doi:10.1109/PGEC.1965.264137
- Duda, R., Hart, P., & Stork, D. (2001). *Pattern classification* (2nd ed.). New York: John Wiley & Sons.
- Everitt, B., Landau, S., & Leese, M. (2001). *Cluster analysis* (4th ed.). London: Arnold.
- Forgy, E. (1965). Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21, 768–780.
- Fritzke, B. (1994). Growing cells structures – a self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7(9), 1441–1460. doi:10.1016/0893-6080(94)90091-4
- Fritzke, B. (1995). A growing neural gas network learns topologies. In G. Tesuaro, D. Touretzky, & T. Leen, T. (Eds.), *Advances in neural information processing systems 7* (pp. 625-632). Cambridge, MA: MIT Press.
- Fritzke, B. (1997). *Some competitive learning methods* (draft document). Retrieved from <http://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/research/gsn/JavaPaper>
- Fukushima, K. (1975). Cognitron: A self-organizing multi-layered neural network. *Biological Cybernetics*, 20, 121–136. doi:10.1007/BF00342633
- Grossberg, S. (1976a). Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23, 121–134. doi:10.1007/BF00344744
- Grossberg, S. (1976b). Adaptive pattern recognition and universal encoding: II. Feedback, expectation, olfaction, and illusions. *Biological Cybernetics*, 23, 187–202. doi:10.1007/BF00344744
- Grossberg, S. (1978). A theory of human memory: Self-organization and performance of sensory-motor codes, maps, and plans. In R. Rosen & F. Snell (Eds.), *Progress in theoretical biology* (Vol. 5). New York: Academic Press.

- Guha, S., Rastogi, R., & Shim, K. (1998). CURE: An efficient clustering algorithm for large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (pp. 73-84).
- Guha, S., Rastogi, R., & Shim, K. (2000). ROCK: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5), 345–366. doi:10.1016/S0306-4379(00)00022-3
- Hammouda, K., & Kamel, M. (2004). Efficient phrase-based document indexing for Web document clustering. *IEEE Transactions on Knowledge and Data Engineering*, 16(10), 1279–1296. doi:10.1109/TKDE.2004.58
- Hansen, P., & Jaumard, B. (1997). Cluster analysis and mathematical programming. *Mathematical Programming*, 79, 191–215.
- Hartigan, J. (1975). *Clustering algorithms*. New York: John Wiley & Sons.
- Haykin, S. (1999). *Neural networks: A comprehensive foundation* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.
- Honkela, T., Kaski, S., Lagus, K., & Kohonen, T. (1997). WEBSOM--self-organizing maps of document collections. In *Proceedings of the Workshop on Self-Organizing Maps – WSOM'97*, Helsinki University of Technology, Neural Networks Research Center, Espoo, Finland (pp. 310-315).
- Huang, J., Georgopoulos, M., & Heileman, G. (1995). Fuzzy ART properties. *Neural Networks*, 8(2), 203–213. doi:10.1016/0893-6080(94)00073-U
- Jain, A., & Dubes, R. (1988). *Algorithms for clustering data*. Englewood Cliffs, NJ: Prentice Hall.
- Jain, A., Murty, M., & Flynn, P. (1999). Data clustering: A review. *ACM Computing Surveys*, 31(3), 264–323. doi:10.1145/331499.331504
- Kaski, S., Honkela, T., Lagus, K., & Kohonen, T. (1998). WEBSOM – self-organizing maps of document collections. *Neurocomputing*, 21, 101–117. doi:10.1016/S0925-2312(98)00039-3
- Kohonen, T. (1989). *Self-organization and associative memory* (3rd ed.). Berlin, Germany: Springer Verlag.
- Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, 78(9), 1464–1480. doi:10.1109/5.58325
- Kohonen, T. (2001). *Self-organizing maps* (3rd ed.), Berlin, Germany: Springer.
- Kohonen, T., Kaski, S., Lagus, K., Salojärvi, J., Honkela, J., Paatero, V., & Saarela, A. (2000). Self organization of a massive document collection. *IEEE Transactions on Neural Networks*, 11(3), 574–585. doi:10.1109/72.846729
- Kondadadi, R., & Kozma, R. (2002). A modified fuzzy ART for soft document clustering. In . *Proceedings of International Joint Conference on Neural Networks, 2002*, 2545–2549.
- Lenstra, J. (1974). Clustering a data array and the traveling salesman problem. *Operations Research*, 22(2), 413–414. doi:10.1287/opre.22.2.413

- Lin, S., & Kernighan, B. (1973). An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21, 498–516. doi:10.1287/opre.21.2.498
- Linares-Barranco, B., Serrano-Gotarredona, M., & Andreaou, A. (1998). *Adaptive resonance theory microchips: Circuit design techniques*. Norwell, MA: Kluwer Academic Publisher.
- MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium*, 1 (pp. 281-297).
- Martinetz, T., Berkovich, S., & Schulten, K. (1993). “Neural-Gas” network for vector quantization and its application to time-series prediction. *IEEE Transactions on Neural Networks*, 4(4), 558–569. doi:10.1109/72.238311
- Martinetz, T., & Schulten, K. (1994). Topology representing networks. *Neural Networks*, 7(3), 507–522. doi:10.1016/0893-6080(94)90109-0
- Massey, L. (2003). On the quality of ART1 text clustering. *Neural Networks*, 16, 771–778. doi:10.1016/S0893-6080(03)00088-1
- McCormick, W., Schweitzer, P., & White, T. (1972). Problem decomposition and data reorganization by a clustering technique. *Operations Research*, 20, 993–1009. doi:10.1287/opre.20.5.993
- McLachlan, G., & Krishnan, T. (1997). *The EM algorithm and extensions*. New York: Wiley.
- Mercer, J. (1909). Functions of positive and negative type, and their connection with the theory of integral equations. *Transactions of the London Philosophical Society (A)*, 209, 415–446. doi:10.1098/rsta.1909.0016
- Moore, B. (1989). ART1 and pattern clustering. In *Proceedings of the 1988 Connectionist Models Summer School* (pp. 174-185).
- Mulder, S., & Wunsch, D. (2003). Million city traveling salesman problem solution by divide and conquer clustering with adaptive resonance neural networks. *Neural Networks*, 16, 827–832. doi:10.1016/S0893-6080(03)00130-8
- Pal, N., Bezdek, J., & Tsao, E. (1993). Generalized clustering networks and Kohonen’s self-organizing scheme. *IEEE Transactions on Neural Networks*, 4(4), 549–557. doi:10.1109/72.238310
- Rangarajan, S., Pboba, V., Balagani, K., Selmic, R., & Iyengar, S. (2004). Adaptive neural network clustering of Web users. *Computer*, 34–40. doi:10.1109/MC.2004.1297299
- Ridella, S., Rovetta, S., & Zunino, R. (1998). Plastic algorithm for adaptive vector quantization. *Neural Computing & Applications*, 7, 37–51. doi:10.1007/BF01413708
- Rosenblatt, F. (1962). *Principles of neurodynamics*. New York: Spartan.
- Rumelhart, D., & Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science*, 9, 75–112.

- Schölkopf, B., Mika, S., Burges, C., Knirsch, P., Müller, K., Rätsch, G., & Smola, A. (1999). Input space versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5), 1000–1017. doi:10.1109/72.788641
- Steinbach, M., Karypis, G., & Kumar, V. (2000). A comparison of document clustering techniques. *Proceedings of KDD Workshop on Text Mining*.
- Vapnik, V. (2000). *The nature of statistical learning theory* (2nd ed.). New York: Springer-Verlag.
- von der Malsburg, C. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14, 85–100. doi:10.1007/BF00288907
- Williamson, J. (1996). Gaussian ARTMAP: A neural network for fast incremental learning of noisy multidimensional maps. *Neural Networks*, 9(5), 881–897. doi:10.1016/0893-6080(95)00115-8
- Wunsch, D., II. (1991). *An optoelectronic learning machine: Invention, experimentation, analysis of first hardware implementation of the ART1 neural network*. Unpublished doctoral dissertation, University of Washington.
- Wunsch, D. II, Caudell, T., Capps, C., Marks, R., & Falk, R. (1993). An optoelectronic implementation of the adaptive resonance neural network. *IEEE Transactions on Neural Networks*, 4(4), 673–684. doi:10.1109/72.238321
- Wunsch, D., II, & Mulder, S. (2004). Evolutionary algorithms, Markov decision processes, adaptive critic designs, and clustering: Commonalities, hybridization, and performance. In *Proceedings of the IEEE International Conference on Intelligent Sensing and Information Processing*.
- Xu, R., & Wunsch, D., II. (2008). *Clustering*. Hoboken, NJ: Wiley / IEEE Press.
- Zadeh, L. (1965). Fuzzy sets. *Information and Control*, 8, 338–353. doi:10.1016/S0019-9958(65)90241-X
- Zhang, T., Ramakrishnan, R., & Livny, M. (1996) BIRCH: An efficient data clustering method for very large databases. In *Proceedings of the ACM SIGMOD Conference on Management of Data* (pp. 103-114).

KEY TERMS AND DEFINITIONS

Hierarchical Clustering: Hierarchical clustering (HC) organizes data with a sequence of nested partitions, either from singleton clusters to a cluster including all individuals or vice versa.

Partitional Clustering: Partitional clustering directly divides data objects into some pre-specified number of clusters without the hierarchical structure.

Adaptive Resonance Theory: Adaptive resonance theory is a learning theory hypothesizing that resonance in neural circuits can trigger fast learning. It was developed as a solution to the stability-plasticity dilemma and can learn arbitrary input patterns in a stable, fast, and self-organizing way.

Self-Organizing Feature Maps: Self-organizing feature maps belong to the category of soft competitive learning clustering algorithms and aim to represent high-dimensional input patterns with prototype

vectors that can be visualized in a two-dimensional lattice structure or one-dimensional linear structure, while preserving the proximity relationships of the original data as much as possible.

Neural Gas: Neural gas belongs to the class of self-organizing neural networks and is capable of adaptively determining the updating of the neighborhood by using a neighborhood ranking of the prototype vectors within the input space.

Traveling Salesman Problem: Given a complete undirected graph G , where each edge between a pair of vertices is weighted with a non-negative integer cost, the common form of the travelling salesman problem is equivalent to finding the shortest Hamiltonian cycle, which is a tour over G that begins and ends at the same vertex and visits other vertices exactly once.

Group Technology: Group technology refers to the study and implementation of information retrieval systems that can retrieve, store, and compare designs.

Document Clustering: Document clustering is the organization of a large amount of text documents into a small number of meaningful clusters, where each cluster represents a specific topic.

Chapter 2

Principal Graphs and Manifolds

Alexander N. Gorban
University of Leicester, UK

Andrei Y. Zinov'yev
Institut Curie, Paris, France

ABSTRACT

In many physical, statistical, biological and other investigations it is desirable to approximate a system of points by objects of lower dimension and/or complexity. For this purpose, Karl Pearson invented principal component analysis in 1901 and found ‘lines and planes of closest fit to system of points’. The famous k-means algorithm solves the approximation problem too, but by finite sets instead of lines and planes. This chapter gives a brief practical introduction into the methods of construction of general principal objects (i.e., objects embedded in the ‘middle’ of the multidimensional data set). As a basis, the unifying framework of mean squared distance approximation of finite datasets is selected. Principal graphs and manifolds are constructed as generalisations of principal components and k-means principal points. For this purpose, the family of expectation/maximisation algorithms with nearest generalisations is presented. Construction of principal graphs with controlled complexity is based on the graph grammar approach.

INTRODUCTION

In many fields of science, one meets with multivariate (multidimensional) distributions of vectors representing some observations. These distributions are often difficult to analyse and make sense of due to the very nature of human brain which is able to visually manipulate only with the objects of dimension no more than three.

This makes actual the problem of approximating the multidimensional vector distributions by objects of lower dimension and/or complexity while retaining the most important information and structures contained in the initial full and complex data point cloud.

DOI: 10.4018/978-1-60566-766-9.ch002

The most trivial and coarse approximation is collapsing the whole set of vectors into its *mean* point. The mean point represents the ‘most typical’ properties of the system, completely forgetting variability of observations.

The notion of the mean point can be generalized for approximating data by more complex types of objects. In 1901 Pearson proposed to approximate multivariate distributions by *lines* and *planes* (Pearson, 1901). In this way the Principal Component Analysis (PCA) was invented, nowadays a basic statistical tool. Principal lines and planes go through the ‘middle’ of multivariate data distribution and correspond to the first few modes of the multivariate Gaussian distribution approximating the data.

Starting from 1950s (Steinhaus, 1956; Lloyd, 1957; and MacQueen, 1967), it was proposed to approximate the complex multidimensional dataset by several ‘mean’ points. Thus *k-means algorithm* was suggested and nowadays it is one of the most used *clustering methods* in machine learning (see a review presented by Xu & Wunsch, 2008).

Both these directions (PCA and K-Means) were further developed during last decades following two major directions: 1) linear manifolds were generalised for non-linear ones (in simple words, initial lines and planes were bended and twisted), and 2) some links between the ‘mean’ points were introduced. This led to the appearance of several large families of new statistical methods; the most famous from them are Principal Curves, Principal Manifolds and Self-Organising Maps (SOM). It was quickly realized that the objects that are constructed by these methods are tightly connected theoretically. This observation allows now to develop a common framework called “Construction of Principal Objects”. The geometrical nature of these objects can be very different but all of them serve as *data approximators of controllable complexity*. It allows using them in the tasks of *dimension* and *complexity reduction*. In Machine Learning this direction is connected with terms ‘Unsupervised Learning’ and ‘Manifold Learning.’

In this chapter we will overview the major directions in the field of principal objects construction. We will formulate the problem and the classical approaches such as PCA and *k*-means in a unifying framework, and show how it is naturally generalised for the Principal Graphs and Manifolds and the most general types of principal objects, Principal Cubic Complexes. We will systematically introduce the most used ideas and algorithms developed in this field.

Approximations of Finite Datasets

Definition. *Dataset* is a finite set X of objects representing N multivariate (multidimensional) observations. These objects $\mathbf{x}^i \in X$, $i = 1 \dots N$, are embedded in \mathbf{R}^m and in the case of complete data are vectors $\mathbf{x}^i \in \mathbf{R}^m$. We will also refer to the individual components of \mathbf{x}^i as x_k^i such that $\mathbf{x}^i = (x_1^i, x_2^i, \dots, x_m^i)$; we can also represent dataset as a *data matrix* $X = \{x_j^i\}$.

Definition. *Distance function* $\text{dist}(\mathbf{x}, \mathbf{y})$ is defined for any pair of objects \mathbf{x}, \mathbf{y} from X such that three usual axioms are satisfied: $\text{dist}(\mathbf{x}, \mathbf{x}) = 0$, $\text{dist}(\mathbf{x}, \mathbf{y}) = \text{dist}(\mathbf{y}, \mathbf{x})$, $\text{dist}(\mathbf{x}, \mathbf{y}) + \text{dist}(\mathbf{y}, \mathbf{z}) \leq \text{dist}(\mathbf{x}, \mathbf{z})$.

Definition. *Mean point* $\mathbf{M}_F(X)$ for X is a vector $\mathbf{M}_F \in \mathbf{R}^m$ such that $\mathbf{M}_F(X) = \arg \min_{\mathbf{y} \in \mathbf{R}^m} \sum_{i=1..N} (\text{dist}(\mathbf{y}, \mathbf{x}_i))^2$.

In this form the definition of the mean point goes back to Fréchet (1948). Notice that in this definition the mean point by Fréchet can be non-unique. However, this definition allows multiple useful generalisations including using it in the abstract metric spaces. It is easy to show that in the case of complete

data and the Euclidean distance function $\text{dist}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^m (\mathbf{x}^i - \mathbf{y}^i)^2}$, or, more generally, in the case of any quadratic distance function (for example, Mahalanobis distance), the mean point is the standard expectation $M_F(X) = \frac{1}{N} \sum_{i=1}^N \mathbf{x}^i \equiv \mathbf{E}(X)$.

Definition. *Orthogonal projection* $P(\mathbf{x}, Y)$ (generalised) is defined for an object \mathbf{x} and a set (not necessarily finite) of vectors Y as a vector in Y such that $P(\mathbf{x}, Y) = \arg \min_{\mathbf{y} \in Y} \text{dist}(\mathbf{x}, \mathbf{y})$. Notice that in principle, one can have non-unique and even infinitely many projections of \mathbf{x} on Y .

Definition. *Mean squared distance* $\text{MSD}(X, Y)$ between a dataset X and a set of vectors Y is defined as $\text{MSD}(X, Y) = \sqrt{\frac{1}{N} \sum_{i=1}^N \text{dist}^2(\mathbf{x}^i, P(\mathbf{x}^i, Y))}$. We will also consider a simple generalisation of MSD : *weighted mean squared distance* $\text{MSD}_W(X, Y) = \sqrt{\frac{1}{\sum_{i=1}^N w_i} \cdot \sum_{i=1}^N w_i \text{dist}^2(\mathbf{x}^i, P(\mathbf{x}^i, Y))}$, where $w_i > 0$ is a weight for the object \mathbf{x}_i .

Our objective in the rest of the chapter is to briefly describe the methods for constructing various approximations (principal objects) for a dataset X . In almost all cases the principal objects will be represented as a finite or infinite set of vectors $Y \in \mathbb{R}^m$ such that 1) it approximates the finite dataset X in the sense of minimisation of $\text{MSD}(X, Y)$, and 2) it answers some *regularity conditions* that will be discussed below.

Probabilistic Interpretation of Statistics and Notion of Self-Consistency

In his original works, Pearson followed the principle that the only reality in data analysis is the dataset, embedded in a multidimensional metric space. This approach can be called *geometrical*. During the 20th century, probabilistic interpretation of statistics was actively developed. Accordingly to this interpretation, a dataset X is one particular of i.i.d. sample from a multidimensional probability distribution $F(\mathbf{x})$ which defines a probability of appearance of a sample in the point $\mathbf{x} \in \mathbb{R}^m$.

The probability distribution, if can be estimated, provides a very useful auxiliary object allowing to define many notions in the theory of statistical data analysis. In particular, it allows us to define principal manifolds as self-consistent objects.

The notion of self-consistency in this context was first introduced by Efron (1967) and developed in the works of Flury (Tarpey & Flury, 1996), where it is claimed to be one of the most fundamental in statistical theory.

Definition. Given probability distribution $F(\mathbf{x})$ and a set of vectors Y we say that Y is *self-consistent with respect to $F(\mathbf{x})$* if $\mathbf{y} = \mathbf{E}_F(\mathbf{x} | P(\mathbf{x}, Y) = \mathbf{y})$ for every vector $\mathbf{y} \in Y$. In words, it means that any vector $\mathbf{y} \in Y$ is a conditional mean expectation of point \mathbf{x} under condition that \mathbf{x} is orthogonally projected in \mathbf{y} .

The disadvantage of this definition for finite datasets is that it is not always possible to calculate the conditional mean, since typically for points $\mathbf{y} \in Y$ it is only one or zero point projected from X . This means that for finite datasets we should develop *coarse-grained self-consistency* notion. Usually it means that for every point $\mathbf{y} \in Y$ one defines some kind of neighbourhood and introduces a modified self-consistency

with respect to this neighbourhood instead of \mathbf{y} itself. Concrete implementations of this idea are described further in this chapter. In all cases, the effective size of the neighbourhood is a fundamental parameter in *controlling the complexity* of the resulting approximator Y .

Four Approaches to Classical PCA

We can define linear principal manifolds as mean squared distance data approximators, constructed from linear manifolds embedded in \mathbf{R}^m . In fact, this corresponds to the original definition of principal lines and planes by Pearson (Pearson, 1901). However, PCA method was re-invented in other fields and even obtained different names (Karhunen-Loëve or KL decomposition (Karhunen, 1946; Loëve, 1955), Hotelling transform (Hotelling, 1933), Proper Orthogonal Decomposition (Lumley, 1967)) and others. Here we formulate four equivalent ways to define principal components that the user can meet in different applications.

Let us consider a linear manifold L_k of dimension k in the parametric form $L_k = \{\mathbf{a}_0 + \beta_1 \mathbf{a}_1 + \dots + \beta_k \mathbf{a}_k \mid \beta_i \in \mathbf{R}\}$, $\mathbf{a}_0 \in \mathbf{R}^m$ and $\{\mathbf{a}_1, \dots, \mathbf{a}_k\}$ is a set of orthonormal vectors in \mathbf{R}^m .

Definition of PCA problem #1 (*data approximation by lines and planes*): PCA problem consists in finding such sequence L_k ($k=1,2,\dots,m-1$) that the sum of squared distances from data points to their orthogonal projections on L_k is minimal over all linear manifolds of dimension k embedded in \mathbf{R}^m : $\text{MSD}(X, L_k) \rightarrow \min$ ($k=1,2,\dots,m-1$).

Definition of PCA problem #2 (*variance maximisation*): For a set of vectors X and for a given \mathbf{a}_i , let us construct a one-dimensional distribution $B^i = \{\beta: \beta = (\mathbf{x}, \mathbf{a}_i), \mathbf{x} \in X\}$ where (\cdot, \cdot) denotes scalar vector product. Then let us define empirical variance of X along \mathbf{a}_i as $\text{Var}(B^i)$, where $\text{Var}()$ is the standard empirical variance. PCA problem consists in finding such L_k that the sum of empirical variances of X along $\mathbf{a}_1, \dots, \mathbf{a}_k$ would be maximal over all linear manifolds of dimension k embedded in \mathbf{R}^m : $\sum_{i=1..k} \text{Var}(B^i) \rightarrow \max$.

Let us also consider an orthogonal complement $\{\mathbf{a}_{k+1}, \dots, \mathbf{a}_m\}$ of the basis $\{\mathbf{a}_1, \dots, \mathbf{a}_k\}$. Then an equivalent definition (*minimization of residue variance*) is

$$\sum_{i=k+1}^m \text{Var}(B^i) \rightarrow \min$$

Definition of PCA problem #3 (*mean point-to-point squared distance maximisation*): PCA problem consists in finding such sequence L_k that the mean point-to-point squared distance between the orthogonal projections of data points on L_k is maximal over all linear manifolds of dimension k embedded in

\mathbf{R}^m : $\frac{1}{N} \sum_{i,j=1}^N \text{dist}^2(P(\mathbf{x}^i, L_k), P(\mathbf{x}^j, L_k)) \rightarrow \max$. Having in mind that all orthogonal projections onto

lower-dimensional space lead to contraction of all point-to-point distances (except for some that do not change), this is equivalent to minimisation of *mean squared distance distortion*:

$$\sum_{i,j=1}^N [\text{dist}^2(\mathbf{x}^i, \mathbf{x}^j) - \text{dist}^2(P(\mathbf{x}^i, L_k), P(\mathbf{x}^j, L_k))] \rightarrow \min$$

In the three above mentioned definitions, the basis vectors are defined up to an arbitrary rotation that does not change the manifold. To make the choice less ambiguous, in the PCA method the following principle is applied: given $\{\mathbf{a}_0, \mathbf{a}_1, \dots, \mathbf{a}_k\}$, any ‘embedded’ linear manifold of smaller dimension s in the form $L_s = \{\mathbf{a}_0 + \beta_1 \mathbf{a}_1 + \dots + \beta_s \mathbf{a}_s \mid \beta_i \in \mathbb{R}, s < k\}$, must be itself a linear principal manifold of dimension s for X (a *flag* of principal subspaces).

Definition of PCA problem #4 (*correlation cancellation*): Find such an orthonormal basis $(\mathbf{a}_1, \dots, \mathbf{a}_s)$ in which the covariance matrix for \mathbf{x} is diagonal. Evidently, in this basis the distributions $(\mathbf{a}_i, \mathbf{x})$ and $(\mathbf{a}_j, \mathbf{x})$, for $i \neq j$, have zero correlation.

Definitions 1-3 were given for finite datasets while definition 4 is sensible both for finite datasets and random vector \mathbf{x} . For finite datasets the empiric correlation should be cancelled. The empiric principal components which annul empiric correlations could be considered as an approximation to the principal components of the random vector.

Equivalence of the above-mentioned definitions in the case of complete data and Euclidean space follows from Pythagorean Theorem and elementary algebra. However, in practice this or that definition can be more useful for computations or generalisations of the PCA approach. Thus, only definitions #1 and #3 are suitable for working with incomplete data since they are defined with use of only distance function that can be easily calculated for the ‘gapped’ data vectors (see further). The definition #1 can be generalized by weighting data points (Cochran & Horne, 1977), while the definition #3 can be generalized by weighting pairs of data points (Gabriel & Zamir, 1979). More details about PCA and generalisations could be found in the fundamental book by Jolliffe (2002).

Basic Expectation/Maximisation Iterative Algorithm for Finding Principal Objects

Most of the algorithms for finding principal objects for a given dataset X are constructed accordingly to the classical expectation/maximisation (EM) splitting scheme that was first formulated as a generic method by Dempster et al (1977):

Generic Expectation-Maximisation algorithm for estimating principal objects

- 1) *Initialisation step.* Some initial configuration of the principal object Y is generated;
- 2) *Expectation (projection) step.* Given configuration of Y , calculate orthogonal projections $P(\mathbf{x}, Y)$, for all $\mathbf{x} \in X$;
- 3) *Maximisation step.* Given the calculated projections, find more optimal configuration of Y with respect to X .
- 4) *(Optional) adaptation step.* Using some strategy, change the properties of Y (typically, add or remove points to Y).
- 5) Repeat steps 2-4 until some convergence criteria would be satisfied.

For example, for the principal line, we have the following implementation of the above mentioned bi-iteration scheme (Bauer, 1957; for generalisations see works of Roweis (1998) and Gorban & Rossiev (1999)).

Iterative algorithm for calculating the first principal component

- 1) Set $\mathbf{a}_0 = \mathbf{M}_F(X)$ (i.e., zero order principal component is the mean point of X);
- 2) Choose randomly \mathbf{a}_1 ;
- 3) Calculate $b_i = \frac{(\mathbf{x}_i - \mathbf{a}_0, \mathbf{a}_1)}{\|\mathbf{a}_1\|^2}$, $i = 1 \dots N$;
- 4) Given b_i , find new \mathbf{a}_1 , such that $\sum_{i=1}^N (\mathbf{x}_i - \mathbf{a}_0 - \mathbf{a}_1 b_i)^2 \rightarrow \min$, i.e. $\mathbf{a}_1 = \frac{\sum_{i=1..N} \mathbf{x}_i b_i - \mathbf{a}_0 \sum_{i=1..N} b_i}{\sum_{i=1..N} b_i^2}$;
- 5) Re-normalize $\mathbf{a}_1 := \mathbf{a}_1 / \|\mathbf{a}_1\|$.
- 6) Repeat steps 3-5 until the direction of \mathbf{a}_1 do not change more than on some small angle ε .

Remark. To calculate all other principal components, *deflation* approach is applied: after finding \mathbf{a}_1 , one calculates new $X^{(1)} = X - \mathbf{a}_0 - \mathbf{a}_1(\mathbf{x}, \mathbf{a}_1)$, and the procedure is repeated for $X^{(1)}$.

Remark. The basic EM procedure has good convergence properties only if the first eigenvalues of the empirical covariance matrix $X^T X$ are sufficiently well separated. If this is not the case, more sophisticated approaches are needed (Bau & Trefethen, 1997).

The PCA method can be treated as *spectral decomposition* of the symmetric and positively defined empirical covariance data matrix (defined in the case of complete data) $C = \frac{1}{N-1} X^T X$ or $C_{ij} = \frac{1}{N-1} \sum_{k=1}^N x_i^k x_j^k$, where without loss of generality we suppose that the data are centered.

Definition. We call $\sigma > 0$ a *singular value* for the data matrix X iff there exist two vectors of unit length \mathbf{a}_σ and \mathbf{b}_σ such that $X \mathbf{a}_\sigma = \sigma \mathbf{b}_\sigma^T$ and $\mathbf{b}_\sigma X = \sigma \mathbf{a}_\sigma^T$. Then the vectors $\mathbf{a}_\sigma = \{a_1^{(\sigma)}, \dots, a_m^{(\sigma)}\}$ and $\mathbf{b}_\sigma = \{b_1^{(\sigma)}, \dots, b_N^{(\sigma)}\}$ are called *left and right singular vectors for the singular value σ* .

If we know all p singular values of X , where $p = \text{rank}(X) \leq \min(N, m)$, then we can represent X as

$$X = \sum_{l=1}^p \sigma_l \mathbf{b}_{(l)} \mathbf{a}_{(l)} \quad \text{or} \quad x_i^k = \sum_{l=1}^p \sigma_l b_k^{(l)} a_i^{(l)}$$
. It is called the *singular value decomposition* (SVD) of X . It is easy to check that the vectors $\mathbf{a}_{(l)}$ correspond to the principal vectors of X and the eigenvectors of the empirical covariance matrix C , whereas $\mathbf{b}_{(l)}$ contain projections of N points onto the corresponding principal vector. Eigenvalues λ_l of C and singular values σ_l of X are connected by $\lambda_l = \frac{1}{N-1} (\sigma_l)^2$.

The mathematical basis for SVD was introduced by Sylvester (1889) and it represents a solid mathematical foundation for PCA (Strang, 1993). Although formally the problems of spectral decomposition of X and eigen decomposition of C are equivalent, the algorithms for performing singular decomposition directly (without explicit calculation of C) can be more efficient and robust (Bau III & Trefethen, 1997). Thus, the iterative EM algorithm for calculating the first principal component described in the previous chapter indeed performs singular decomposition (for centered data we simply put $\mathbf{a}_0 = \mathbf{0}$) and finds right singular (principal) and left singular vectors one by one.

K-Means and Principal Points

K-means clustering goes back to 1950s (Steinhaus (1956); Lloyd (1957); and MacQueen (1967)). It is another extreme in its simplicity case of finding a principal object. In this case it is simply an unstructured

finite (and usually, much smaller than the number of points N in the dataset X) set of vectors (centroids). One can say that the solution searched by the k -means algorithm is a set of k principal points (Flury, 1990).

Definition. A set of k points $Y=\{\mathbf{y}_1, \dots, \mathbf{y}_k\}$, $\mathbf{y}_i \in \mathbb{R}^m$ is called a *set of principal points* for dataset X if it approximates X with minimal mean squared distance error over all sets of k -points in \mathbb{R}^m (distortion): $\sum_{\mathbf{x} \in X} \text{dist}^2(\mathbf{x}, P(\mathbf{x}, Y)) \rightarrow \min$, where $P(\mathbf{x}, Y)$ is the point from Y closest to \mathbf{x} . Note that the set of principal points can be not unique.

The simplest implementation of the k -means procedure follows the classical EM scheme:

Basic k -means algorithm

- 1) Choose initial position of $\mathbf{y}_1, \dots, \mathbf{y}_k$ randomly from $\mathbf{x}_i \in X$ (with equal probabilities);
- 2) Partition X into subsets K_i , $i=1..k$ of data points by their proximity to \mathbf{y}_k :

$$K_i = \{\mathbf{x} : \mathbf{y}_i = \arg \min_{\mathbf{y}_j \in Y} \text{dist}(\mathbf{x}, \mathbf{y}_j)\};$$
- 3) Re-estimate $\mathbf{y}_i = \frac{1}{|K_i|} \sum_{\mathbf{x} \in K_i} \mathbf{x}$, $i = 1..k$;
- 4) Repeat steps 2-3 until complete convergence.

The method is sensitive to the initial choice of $\mathbf{y}_1, \dots, \mathbf{y}_k$. Arthur & Vassilvitskii (2007) demonstrated that the special construction of probabilities instead of equidistribution gives serious advantages. The first centre, \mathbf{y}_1 , they select equiprobable from X . Let the centres $\mathbf{y}_1, \dots, \mathbf{y}_j$ are chosen ($j < k$) and $D(\mathbf{x})$ be the squared shortest distance from a data point \mathbf{x} to the closest centre we have already chosen. Then, we select the next centre, \mathbf{y}_{j+1} , from $\mathbf{x}_i \in X$ with probability

$$p(\mathbf{x}_i) = D(\mathbf{x}_i) / \sum_{\mathbf{x} \in X} D(\mathbf{x}).$$

Evidently, any solution of k -means procedure converges to a self-consistent set of points $Y=\{\mathbf{y}_1, \dots, \mathbf{y}_k\}$ (because $Y = E[P(X, Y)]$), but this solution may give a local minimum of distortion and is not necessary the set of principal points (which is the globally optimal approximator from all possible k -means solutions).

Multiple generalisations of k -means scheme have been developed (see, for example, a book of Mirkin (2005) based on the idea of ‘data recovering’). The most computationally expensive step of the algorithm, partitioning the dataset by proximity to the centroids, can be significantly accelerated using kd -tree data structure (Pelleg & Moore, 1999). Analysis of the effectiveness of EM algorithm for the k -means problem was given by Ostrovsky et al. (2006).

Notice that the case of principal points is the only in this chapter when self-consistency and coarse-grained self-consistency coincide: centroid \mathbf{y}_k is the conditional mean point for the data points belonging to the Voronoi region associated with \mathbf{y}_k .

Local PCA

The term ‘Local PCA’ was first used by Braverman (1970) and Fukunaga & Olsen (1971) to denote the simplest cluster-wise PCA approach which consists in 1) applying k -means or other type of clustering to a dataset and 2) calculating the principal components for each cluster separately. However, this simple idea performs rather poorly in applications, and more interesting approach consists in generalizing k -means by introducing *principal hyperplane segments* proposed by Diday (1979) and called ‘ k -segments’ or *local subspace analysis* in a more advanced version (Liu, 2003). The algorithm for their estimation follows the classical EM scheme.

Further development of the local PCA idea went in two main directions. First, Verbeek (2002) proposed a variant of the ‘ k -segment’ approach for one-dimensional segments accompanied by a strategy to assemble disconnected line segments into the global piecewise linear principal curve. Einbeck et al (2008) proposed an iterative cluster splitting and joining approach (*recursive local PCA*) which helps to select the optimal number and configuration of disjoined segments.

Second direction is associated with a different understanding of ‘locality’. It consists in calculating *local mean points* and *local principal directions* and following them starting from (may be multiple) seed points. Locality is introduced using kernel functions defining the effective radius of neighborhood in the data space. Thus, Delicado (2001) introduced *principal oriented points* (POP) based on the variance maximisation-based definition of PCA (#2 in our chapter). POPs are different from the principal points introduced above because they are defined independently one from another, while the principal points are defined globally, as a set. POPs can be assembled into the *principal curves of oriented points* (PCOP). Einbeck (2005) proposed a simpler approach based on local tracing of principal curves by calculating *local centers of mass* and the *local first principal components*.

SOM Approach for Principal Manifold Approximation and its Generalisations

Kohonen in his seminal paper (Kohonen, 1982) proposed to modify the k -means approach by introducing connections between centroids such that a change in the position of one centroid would also change the configuration of some neighboring centroids. Thus Self-Organizing Maps (SOM) algorithm was developed.

With the SOM algorithm (Kohonen, 1982) we take a finite metric space V with metric ρ and try to map it into \mathbf{R}^m with combinations of two criteria: (1) the best preservation of initial structure in the image of V and (2) the best approximation of the dataset X . In this way, SOMs give the most popular approximations for principal manifolds: we can take for V a fragment of a regular s -dimensional grid and consider the resulting SOM as the approximation to the s -dimensional principal manifold (Mulier & Cherkassky, 1995; Ritter et al, 1992; Yin H. 2008).

The SOM algorithm has several setup variables to regulate the compromise between these goals. In the original formulation by Kohonen, we start from some initial approximation of the map, $\phi_1: V \rightarrow \mathbf{R}^m$. Usually this approximation lies on the s -dimensional linear principal manifold. On each k -th step of the algorithm we have a chosen datapoint $\mathbf{x} \in X$ and a current approximation $\phi_k: V \rightarrow \mathbf{R}^m$. For these \mathbf{x} and ϕ_k we define an ‘owner’ of \mathbf{x} in V : $v_x = \arg \min_{v \in V} \|\mathbf{x} - \varphi_k(v)\|$. The next approximation, ϕ_{k+1} , is $\phi_{k+1}(v) = h_k \times w(\rho(v, v_x))(\mathbf{x} - \phi_k(v))$. Here h_k is a step size, $0 \leq w(\rho(v, v_x)) \leq 1$ is a monotonically decreasing neighborhood function. This process proceeds in several epochs, with neighborhood radius decreasing

during each next epoch.

The idea of SOM is flexible, was applied in many domains of science, and it lead to multiple generalizations (see the review paper by Yin (2008)). Some of the algorithms for constructing SOMs are of EM type described above, such as the *Batch SOM Algorithm* (Kohonen, 1997): it includes projecting step exactly the same as in k -means and the maximization step at which all $\phi_k(v)$ are modified simultaneously.

One source of theoretical dissatisfaction with SOM is that it is not possible to define an optimality criterion (Erwin et al, 1992): SOM is a result of the algorithm at work and there does not exist any objective function that is minimized by the training process.

In attempt to resolve this issue, Bishop et al. (1998) developed the optimization-based Generative Topographic Mapping (GTM) method. In this setting, it is supposed that the observed data is i.i.d. sample from a mixture of Gaussian distributions with the centers aligned along a two-dimensional grid, embedded in the data space. Parameters of this mixture are determined by EM-based maximization of the likelihood function (probability of observing X within this data model).

Principal Manifolds by Hastie and Stuelze

Principal curves and principal two-dimensional surfaces for a probability distribution $F(\mathbf{x})$ were introduced in the PhD thesis by Trevor Hastie (1984) as a self-consistent (non-linear) one- and two-dimensional globally parametrisable smooth manifolds without self-intersections.

Definition. Let G be the class of differentiable 1-dimensional curves in \mathbf{R}^m , parameterized by $\lambda \in \mathbf{R}^1$ and without self-intersections. The *Principal Curve* of the probability distribution $F(\mathbf{x})$ is such a $Y(\lambda) \in G$ that is self-consistent.

Remark. Usually, a compact subset of \mathbf{R}^m and a compact interval of parameters $\lambda \in \mathbf{R}^1$ are considered. To discuss unbounded regions, it is necessary to add a condition that $Y(\lambda)$ has finite length inside any bounded subset of \mathbf{R}^m (Kégl, 1999).

Definition. Let G^2 be the class of differentiable 2-dimensional surfaces in \mathbf{R}^m , parameterized by $\lambda \in \mathbf{R}^2$ and without self-intersections. The *Principal Surface* of the probability distribution $F(\mathbf{x})$ is such a $Y(\lambda) \in G^2$ that is self-consistent. (Again, for unbounded regions it is necessary to assume that for any bounded set B from \mathbf{R}^m the set of parameters λ for which $Y(\lambda) \in B$ is also bounded.)

First, Hastie and Stuelze proposed an algorithm for finding the principal curves and principal surfaces for a probability distribution $F(\mathbf{x})$, using the classical EM splitting. We do not provide this algorithm here because for a finite dataset X it can not be directly applied because in a typical point on $Y(\lambda)$ only zero or one data point is projected, hence, one can not calculate the expectation. As mentioned above, in this case we should use some kind of coarse-grained self-consistency. In the original approach by Hastie (1984), this is done through introducing *smoothers*. This gives the practical formulation of the HS algorithm for estimating the principal manifolds from a finite dataset X :

Hastie and Stuelze algorithm for finding principal curve for finite dataset

- 1) Initialize $Y(\lambda) = \mathbf{a}_0 + \lambda \mathbf{a}_1$, where \mathbf{a}_0 is a mean point and \mathbf{a}_1 is the first principal component;

- 2) Project every data point \mathbf{x}_i onto $Y(\lambda)$: i.e., for each \mathbf{x}_i find λ_i such that $Y(\lambda_i) = \arg \inf_{\lambda} \| Y(\lambda) - \mathbf{x}_i \|^2$. In practice it requires interpolation procedure because $Y(\lambda)$ is determined in a finite number of points $\{\lambda_1, \dots, \lambda_N\}$. The simplest is the piecewise interpolation procedure, but more sophisticated procedures can be proposed (Hastie, 1984);
- 3) Calculate new $Y'(\lambda)$ in the finite number of internal coordinates $\{\lambda_1, \dots, \lambda_N\}$ (found at the previous step) as the local average of points \mathbf{x}_i and some other points, that have close to λ_i projections onto Y . To do this, 1) a *span* $[w \times N]$ is defined ([.] here is integer part), where $0 < w \ll 1$ is a parameter of the method (coarse-grained self-consistency neighbourhood radius); 2) for $[w \times N]$ internal coordinates $\{\lambda_{i_1}, \dots, \lambda_{i_{[w \times N]}}\}$ closest to λ_i and the corresponding $\{\mathbf{x}_{i_1}, \dots, \mathbf{x}_{i_{[w \times N]}}\}$ calculate weighted least squares linear regression $y(\lambda) = \mathbf{a}^{(i)}\lambda + \mathbf{b}^{(i)}$; 3) define $Y'(\lambda_i)$ as the value of the linear regression in λ_i : $Y'(\lambda_i) = \mathbf{a}^{(i)}\lambda_i + \mathbf{b}^{(i)}$.
- 4) Reassign $Y(\lambda) \leftarrow Y'(\lambda)$
- 5) Repeat steps 2)-4) until Y does not change (approximately).

Remark. For the weights in the regression at the step 3) Hastie proposed to use some symmetric kernel function that vanishes on the borders of the neighbourhood. For example, for \mathbf{x}_i let us denote as $\lambda_{i_{[w \times N]}}$ the most distant value of the internal coordinate from $[w \times N]$ ones closest to λ_i . Then we can define weight for the pair $(\lambda_{i_j}, \mathbf{x}_{i_j})$ as

$$\omega_j^i = \begin{cases} (1 - (|\lambda_{i_j} - \lambda_i| / |\lambda_{i_j} - \lambda_{i_N}|)^3)^{1/3}, & \text{if } |\lambda_{i_j} - \lambda_i| < |\lambda_{i_j} - \lambda_{i_N}|, \\ 0, & \text{otherwise.} \end{cases}$$

Remark. At the step 3) an alternative approach was also proposed with use of cubic splines to approximate the smooth function $Y(\lambda)$ from all pairs $(\lambda_i, \mathbf{x}_i)$, $i = 1..N$.

Non-linear Principal Manifolds constructed by this algorithm are usually called *Hastie-Staelze (HS) principal manifolds*. However, the global optimality of HS principal manifolds is not guaranteed (only self-consistency in the case of distribution or coarse-grained self-consistency in the case of dataset is guaranteed by construction). For example, the second principal component of a sample X from a normal distribution is self-consistent and will be correct HS principal curve but of course not the optimal one.

We should also underline that our view on what is the object constructed by the HS algorithm for a dataset X depends on 1) probabilistic interpretation of the nature of X , and 2) the chosen heuristic approach to coarse-grained self-consistency. If we do not suppose that the dataset is generated by i.i.d. sampling from $F(\mathbf{x})$ then the definition of HS principal manifold is purely operational: HS principal manifold for X is the result of application of HS algorithm for finite datasets. Analogous remark is applicable for all principal manifold approximators constructed for finite datasets and described further in this chapter.

In his PhD thesis Hastie noticed that the HS principal curve does not coincide with the generating curve in a very simple additive data generation model

$$X = f(\lambda) + \varepsilon, \tag{1}$$

where $f(\lambda)$ is some curve embedded in data space and ε is noise distribution independent on λ . Because of the fact that if $f(\lambda)$ is not a straight line then it is not self-consistent, HS principal curves were claimed to be ‘biased’. This inspired Tibshirani (1992) to introduce an alternative definition of the principal curve, based directly on a continuous mixture model (1) and maximising regularized likelihood.

Kégl-Kryzhak Improvement

Kégl in his PhD thesis supervised by Kryzhak (Kégl, 1999) revised the existing methods for estimating the principal curves. In particular, this led to the definition of principal curves with limited length.

Definition. *Principal curve $Y_L(\lambda)$ of length L* is such a curve that the mean squared distance from the dataset X to the curve $Y_L(\lambda)$ is minimal over all curves of length less than or equal to L :

$$\sum_{i=1}^N \text{dist}^2(\mathbf{x}^i, P(\mathbf{x}^i, Y_L)) \rightarrow \min$$

Theorem. Assume that X has finite second moments, i.e. $\sum_{i=1}^N \mathbf{x}^i (\mathbf{x}^i)^T < \infty$. Then for any $L > 0$ there exists a principal curve of length L .

Principal curves of length L as defined by Kégl, are globally optimal approximators as opposite to the HS principal curves that are only self-consistent. However, all attempts to construct a practical algorithm for finding globally optimal principal curves of length L were not successful. Instead Kégl developed an efficient heuristic *Polygonal line algorithm* for constructing piecewise linear principal curves.

Let us consider a piecewise curve Y composed from vertices located in points $\{\mathbf{y}^1, \dots, \mathbf{y}^{k+1}\}$ and k segments connecting pairs of vertices $\{\mathbf{y}^j, \mathbf{y}^{j+1}\}$, $j=1..k$. Kégl’s algorithm searches for a (local) optimum of the penalised mean squared distance error function:

$$U(X, Y) = \text{MSD}(X, Y) + \frac{\lambda}{k+1} \sum_{i=1}^{k+1} \text{CP}(i), \quad (2)$$

where $\text{CP}(i)$ is a curvature penalty function for a vertex i chosen as

$$\text{CP}(i) = \begin{cases} \|\mathbf{y}^1 - \mathbf{y}^2\|^2 & \text{if } i = 1 \\ r^2(1 + \cos \gamma(i)) & \text{if } 1 < i < k+1 \\ \|\mathbf{y}^k - \mathbf{y}^{k+1}\|^2 & \text{if } i = k+1 \end{cases},$$

where $\cos \gamma(i) = \frac{(\mathbf{y}^{i-1} - \mathbf{y}^i, \mathbf{y}^{i+1} - \mathbf{y}^i)}{\|\mathbf{y}^{i-1} - \mathbf{y}^i\| \|\mathbf{y}^{i+1} - \mathbf{y}^i\|}$ is the cosines of the angle between two neighbouring

segments at the vertex i , $r = \max_{\mathbf{x} \in X} \text{dist}(\mathbf{x}, \mathbf{M}_F(X))$ is the ‘radius’ of the dataset X , and λ is a parameter controlling the curve global smoothness.

The *Polygonal line algorithm* (Kégl, 1999) follows the standard EM splitting scheme:

Polygonal line algorithm for estimating piece-wise linear principal curve

- 1) The initial approximation is constructed as a segment of principal line. The length of the segment is the difference between the maximal and the minimal projection value of X onto the first principal component. The segment is positioned such that it contains all of the projected data points. Thus in the initial approximation one has two vertices $\{\mathbf{y}^1, \mathbf{y}^2\}$ and one segment between them ($k = 1$).
- 2) *Projection step.* The dataset X is partitioned into $2k+1$ $K_z = \{\mathbf{x} : z = \arg \min_{z \in \text{vertices} \cap \text{segments}} \text{dist}(\mathbf{x}, z)\}$ subsets constructed by their proximity to $k+1$ vertices and k segments. If a segment i and a vertex j are equally distant from \mathbf{x} then \mathbf{x} is placed into K_j only.
- 3) *Optimisation step.* Given partitioning obtained at the step 2, the functional $U(X, Y)$ is optimised by use of a gradient technique. Fixing partitioning into K_i is needed to calculate the gradient of $U(X, Y)$ because otherwise it is not a differentiable function with respect to the position of vertices $\{\mathbf{y}_i\}$.
- 4) *Adaptation step.* Choose the segment with the largest number of points projected onto it. If more than one such segment exists then the longest one is chosen. The new vertex is inserted in the midpoint of this segment; all other segments are renumerated accordingly.
- 5) *Stopping criterion.* The algorithm stops when the number of segments exceeds

$$\beta \cdot N^{1/3} \cdot \frac{r}{\text{MSD}(X, Y)}$$

Heuristically, the default parameters of the method have been proposed $\beta = 0.3$, $\lambda = \lambda' \cdot \frac{k}{N^{1/3}} \cdot \frac{\text{MSD}(X, Y)}{r}$, $\lambda' = 0.13$. The details of implementation together with convergence and computational complexity study are provided elsewhere (Kégl, 1999).

Smola et al. (2001) proposed a *regularized principal manifolds* framework, based on minimization of quantization error functional with a large class of regularizers that can be used and a universal EM-type algorithm. For this algorithm, the convergence rates were analyzed and it was showed that for some regularizing terms the convergence can be optimized with respect to the Kegl's polygonal line algorithm.

Elastic Maps Approach

In a series of works (Gorban & Rossiev, 1999; Gorban et al., 2001, 2003; Gorban & Zinovyev, 2005, 2008a; Gorban et al., 2007, 2008), the authors of this chapter used metaphor of elastic membrane and plate to construct one-, two- and three-dimensional principal manifold approximations of various topologies. Mean squared distance approximation error combined with the elastic energy of the membrane serves as a functional to be optimised. The elastic map algorithm is extremely fast at the optimisation step due to the simplest form of the smoothness penalty. It is implemented in several programming languages as software libraries or front-end user graphical interfaces freely available from the web-site <http://bioinfo.curie.fr/projects/vidaexpert>. The software found applications in microarray data analysis, visualization of genetic texts, visualization of economical and sociological data and other fields (Gorban et al, 2001, 2003; Gorban & Zinovyev 2005, 2008a; Gorban et al, 2007, 2008).

Let G be a simple undirected graph with set of vertices V and set of edges E .

Definition. k -star in a graph G is a subgraph with $k + 1$ vertices $v_{0,1,\dots,k} \in V$ and k edges $\{(v_0, v_i) / i = 1, \dots, k\} \in E$. The rib is by definition a 2-star.

Definition. Suppose that for each $k \geq 2$, a family S_k of k -stars in G has been selected. Then we define an *elastic graph* as a graph with selected families of k -stars S_k and for which for all $E^{(i)} \in E$ and $S_k^{(j)} \in S_k$, the corresponding elasticity moduli $\lambda_i > 0$ and $\mu_{kj} > 0$ are defined.

Definition. *Primitive elastic graph* is an elastic graph in which every non-terminal node (with the number of neighbours more than one) is associated with a k -star formed by *all* neighbours of the node. All k -stars in the primitive elastic graph are selected, i.e. the S_k sets are completely determined by the graph structure.

Definition. Let $E^{(i)}(0), E^{(i)}(1)$ denote two vertices of the graph edge $E^{(i)}$ and $S_k^{(j)}(0), \dots, S_k^{(j)}(k)$ denote vertices of a k -star $S_k^{(j)}$ (where $S_k^{(j)}(0)$ is the central vertex, to which all other vertices are connected). Let us consider a map $\varphi: V \rightarrow \mathbf{R}^m$ which describes an embedding of the graph into a multidimensional space. The *elastic energy of the graph embedding in the Euclidean space* is defined as

$$U^\varphi(G) := U_E^\varphi(G) + U_R^\varphi(G), \quad (3)$$

$$U_E^\varphi(G) := \sum_{E^{(i)}} \lambda_i \left\| \varphi(E^{(i)}(0)) - \varphi(E^{(i)}(1)) \right\|^2, \quad (4)$$

$$U_R^\varphi(G) := \sum_{S_k^{(j)}} \mu_{kj} \left\| \varphi(S_k^{(j)}(0)) - \frac{1}{k} \sum_{i=1}^k \varphi(S_k^{(j)}(i)) \right\|^2. \quad (5)$$

Definition. *Elastic net* is a particular case of elastic graph which (1) contains only ribs (2-stars) (the family S_k are empty for all $k > 2$); and (2) the vertices of this graph form a regular small-dimensional grid (Figure 1).

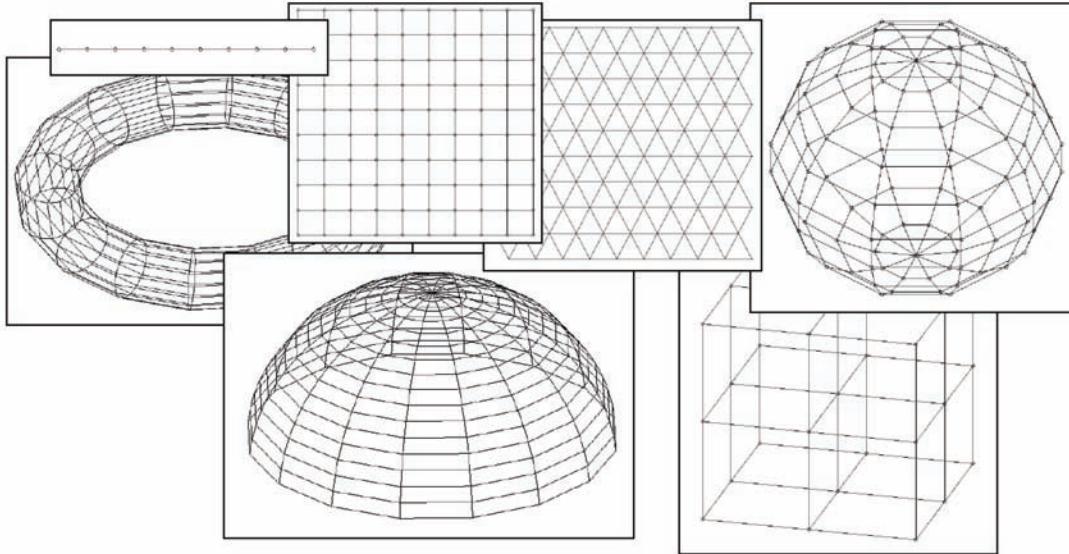
The elastic net is characterised by *internal dimension* $\dim(G)$. Every node v_i in the elastic net is indexed by the discrete values of *internal coordinates* $\{\lambda_1^i, \dots, \lambda_{\dim(G)}^i\}$ in such a way that the nodes close on the graph have similar internal coordinates.

The purpose of the elastic net is to introduce point approximations to manifolds. Historically it was first explored and used in applications. To avoid confusion, one should notice that the term elastic net was independently introduced by several groups: for solving the traveling salesman problem (Durbin & Willshaw, 1987), in the context of principal manifolds (Gorban et al, 2001) and recently in the context of regularized regression problem (Zhou & Hastie, 2005). These three notions are completely independent and denote different things.

Definition. *Elastic map* is a continuous manifold $Y \in \mathbf{R}^m$ constructed from the elastic net as its grid approximation using some between-node interpolation procedure. This interpolation procedure constructs a continuous mapping $\phi_c: \{\lambda_1, \dots, \lambda_{\dim(G)}\} \rightarrow \mathbf{R}^m$ from the discrete map $\varphi: V \rightarrow \mathbf{R}^m$, used to embed the graph in \mathbf{R}^m , and the discrete values of node indices $\{\lambda_1^i, \dots, \lambda_{\dim(G)}^i\}$, $i = 1 \dots |V|$. For example, the simplest *piecewise linear elastic map* is built by piecewise linear map ϕ_c .

Definition. *Elastic principal manifold of dimension s* for a dataset X is an elastic map, constructed from an elastic net Y of dimension s embedded in \mathbf{R}^m using such a map $\phi_{\text{opt}}: Y \rightarrow \mathbf{R}^m$ that corresponds to the minimal value of the functional

Figure 1. Elastic nets used in practice



$$U^\varphi(X, Y) = \text{MSD}_W(X, Y) + U^\varphi(G), \quad (6)$$

where the weighted mean squared distance from the dataset X to the elastic net Y is calculated as the distance to the finite set of vertices $\{\mathbf{y}^1=\phi(v_1), \dots, \mathbf{y}^k=\phi(v_k)\}$.

In the Euclidean space one can apply an EM algorithm for estimating the elastic principal manifold for a finite dataset. It is based in turn on the general algorithm for estimating the locally optimal embedding map ϕ for an arbitrary elastic graph G , described below.

Optimisation of the elastic graph algorithm:

- 1) Choose some initial position of nodes of the elastic graph $\{\mathbf{y}^1=\phi(v_1), \dots, \mathbf{y}^k=\phi(v_k)\}$, where k is the number of graph nodes $k = |V|$;
- 2) Calculate two matrices e_{ij} and s_{ij} , using the following sub-algorithm:
 - i. Initialize the s_{ij} matrix to zero;
 - ii. For each k -star $S_k^{(i)}$ with elasticity module μ_{ki} , outer nodes v_{N1}, \dots, v_{Nk} and the central node v_{N0} , the s_{ij} matrix is updated as follows ($1 \leq l, m \leq k$):

$$\begin{aligned} s_{N_0 N_0} &\leftarrow s_{N_0 N_0} + \mu_{ki}, \quad s_{N_l N_m} \leftarrow s_{N_l N_m} + \mu_{ki}/k^2 \\ s_{N_0 N_l} &\leftarrow s_{N_0 N_l} - \mu_{ki}/k, \quad s_{N_l N_0} \leftarrow s_{N_l N_0} - \mu_{ki}/k \end{aligned}$$
 - iii. Initialize the e_{ij} matrix to zero;
 - iv. For each edge $E^{(i)}$ with weight λ_i , one vertex v_{k1} and the other vertex v_{k2} , the e_{jk} matrix is updated as follows:

$$\begin{aligned} e_{k_1 k_1} &\leftarrow e_{k_1 k_1} + \lambda_i, \quad e_{k_2 k_2} \leftarrow e_{k_2 k_2} + \lambda_i \\ e_{k_1 k_2} &\leftarrow e_{k_1 k_2} - \lambda_i, \quad e_{k_2 k_1} \leftarrow e_{k_2 k_1} - \lambda_i \end{aligned}$$

- 3) Partition X into subsets K_i , $i=1..k$ of data points by their proximity to \mathbf{y}_k :

$$K_i = \{\mathbf{x} : \mathbf{y}_i = \arg \min_{\mathbf{y}_j \in Y} \text{dist}(\mathbf{x}, \mathbf{y}_j)\};$$
- 4) Given K_j , calculate matrix $a_{js} = \frac{n_j \delta_{js}}{\sum_{i=1}^N w_i} + e_{js} + s_{js}$, where $n_j = \sum_{x^i \in K_j} w_i$, δ_{js} is the Kronecker's symbol.
- 5) Find new position of $\{\mathbf{y}^1, \dots, \mathbf{y}^k\}$ by solving the system of linear equations

$$\sum_{s=1}^k a_{js} \mathbf{y}^s = \frac{1}{\sum_{i=1}^N w_i} \cdot \sum_{x^i \in K_j} w_i \mathbf{x}^i$$
- 6) Repeat steps 3-5 until complete or approximate convergence of node positions $\{\mathbf{y}^1, \dots, \mathbf{y}^k\}$.

As usual, the EM algorithm described above gives only locally optimal solution. One can expect that the number of local minima of the energy function U grows with increasing the ‘softness’ of the elastic graph (decreasing μ_{kj} parameters). Because of this, in order to obtain a solution closer to the global optimum, the *softening strategy* has been proposed, used in the algorithm for estimating the elastic principal manifold.

Algorithm for estimating the elastic principal manifold

- 1) Define a decreasing set of numbers $\{m_1, \dots, m_p\}$, $m_p=1$ (for example, $\{10^3, 10^2, 10, 1\}$), defining p epochs for softening;
- 2) Define the base values of the elastic moduli $\lambda_i^{(base)}$ and $\mu_i^{(base)}$;
- 3) Initialize positions of the elastic net nodes $\{\mathbf{y}^1, \dots, \mathbf{y}^k\}$ on the linear principal manifold spanned by first $\dim(G)$ principal components;
- 4) Set $\text{epoch_counter} = 1$
- 5) Set the elastic moduli $\lambda_i = m_{\text{epoch_counter}} \lambda_i^{(base)}$ and $\mu_i = m_{\text{epoch_counter}} \mu_i^{(base)}$;
- 6) Modify the elastic net using the algorithm for optimisation of the elastic graph;
- 7) Repeat steps 5-6 for all values of $\text{epoch_counter} = 2, \dots, p$.

Remark. The values λ_i and μ_i are the coefficients of stretching elasticity of every edge $E^{(i)}$ and of bending elasticity of every rib $S_2^{(j)}$. In the simplest case $\lambda_1 = \lambda_2 = \dots = \lambda_s = \lambda(s)$, $\mu_1 = \mu_2 = \dots = \mu_r = \mu(r)$, where s and r are the numbers of edges and ribs correspondingly. Approximately dependence on graph

‘resolution’ is given by Gorban & Zinov'yev (2007): $\lambda(s) = \lambda_0 \cdot s^{\frac{2-\dim(G)}{\dim(G)}}$, $\mu(s) = \mu_0 \cdot r^{\frac{2-\dim(G)}{\dim(G)}}$. This formula is applicable, of course, only for the elastic nets. In general case λ_i and μ_i are often made variable in different parts of the graph accordingly to some adaptation strategy (Gorban & Zinov'yev, 2005).

Remark. $U_E^\varphi(G)$ penalizes the total length (or, indirectly, ‘square’, ‘volume, etc.) of the constructed manifold and provides regularization of distances between node positions at the initial steps of the softening. At the final stage of the softening λ_i can be put to zero with little effect on the manifold configuration.

Elastic map post-processing such as *map extrapolation* can be applied to increase its usability and avoid the ‘border effect’, for details see (Gorban & Zinov'yev, 2008a).

Pluriharmonic Graphs as Ideal Approximators

Approximating datasets by one dimensional principal curves is not satisfactory in the case of datasets that can be intuitively characterized as *branched*. A principal object which naturally passes through the ‘middle’ of such a data distribution should also have branching points that are missing in the simple structure of principal curves. Introducing such branching points converts principal curves into *principal graphs*.

Principal graphs were introduced by Kégl & Krzyzak (2002) as a natural extension of one-dimensional principal curves in the context of skeletonisation of hand-written symbols. The most important part of this definition is the form of the penalty imposed onto deviation of the configuration of the branching points embedment from their ‘ideal’ configuration (*end*, *line*, *corner*, *T*-, *Y*- and *X*-configuration). Assigning types for all vertices serves for definition of the penalty on the total deviation from the graph ‘ideal’ configuration (Kégl, 1999). Other types of vertices were not considered, and outside the field of symbol skeletonization applicability of such a definition of principal graph remains limited.

Gorban & Zinov'yev (2005), Gorban et al. (2007), and Gorban et al. (2008) proposed to use a universal form of non-linearity penalty for the branching points. The form of this penalty is defined in the previous chapter for the elastic energy of graph embedment. It naturally generalizes the simplest three-point *second derivative* approximation squared:

for a 2-star (or rib) the penalty equals $\|\varphi(S_2^{(j)}(0)) - \frac{1}{2}(\varphi(S_2^{(j)}(1)) + \varphi(S_2^{(j)}(2)))\|^2$,

for a 3-star it is $\|\varphi(S_3^{(j)}(0)) - \frac{1}{3}(\varphi(S_3^{(j)}(1)) + \varphi(S_3^{(j)}(2)) + \varphi(S_3^{(j)}(3)))\|^2$, etc.

For a k -star this penalty equals to zero iff the position of the central node coincides with the mean point of its neighbors. An embedment $\phi(G)$ is ‘ideal’ if all such penalties equal to zero. For a *primitive elastic graph* this means that this embedment is a *harmonic function on graph*: its value in each non-terminal vertex is a mean of the value in the closest neighbors of this vertex.

For non-primitive graphs we can consider stars which include not all neighbors of their centers. For example, for a square lattice we create elastic graph (elastic net) using 2-stars (ribs): all vertical 2-stars and all horizontal 2-stars. For such elastic net, each non-boundary vertex belongs to two stars. For a general elastic graph G with sets of k -stars S_k we introduce the following notion of pluriharmonizing function.

Definition. A map $\phi: V \rightarrow \mathbf{R}^m$ defined on vertices of G is *pluriharmonic* iff for any k -star $S_k^{(j)} \in S_k$ with the central vertex $S_k^{(j)}(0)$ and the neighbouring vertices $S_k^{(j)}(i)$, $i = 1 \dots k$, the equality holds:

$$\varphi(S_k^{(j)}(0)) = \frac{1}{k} \sum_{i=1}^k \varphi(S_k^{(j)}(i)). \quad (7)$$

Pluriharmonic maps generalize the notion of linear map and of harmonic map, simultaneously. For example:

- 1) 1D harmonic functions are linear;
- 2) If we consider an n D cubic lattice as a primitive graph (with $2n$ -stars for all non-boundary vertices), then the correspondent pluriharmonic functions are just harmonic ones;
- 3) If we create from n D cubic lattice a standard n D elastic net with 2-stars (each non-boundary vertex is a center of n 2-stars, one 2-stars for each coordinate direction), then pluriharmonic functions are linear.

Pluriharmonic functions have many attractive properties, for example, they satisfy the following *maximum principle*. A vertex v of an elastic graph is called a *corner point* or an *extreme point* of G iff v is not a centre of any k -star from S_k for all $k > 0$.

Theorem. Let $\phi: V \rightarrow \mathbf{R}^m$ be a pluriharmonic map, F be a convex function on \mathbf{R}^m , and $a = \max_{x \in V} F(\phi(x))$. Then there is a corner point v of G such that $F(\phi(v)) = a$.

Convex functions achieve their maxima in corner points. Even a particular case of this theorem with linear functions F is quite useful. Linear functions achieve their maxima and minima in corner points.

In the theory of principal curves and manifolds the penalty functions were introduced to penalise deviation from linear manifolds (straight lines or planes). We proposed to use pluriharmonic embeddings ('pluriharmonic graphs') as 'ideal objects' instead of manifolds and to introduce penalty (5) for deviation from this ideal form.

Graph Grammars and Three Types of Complexity for Principal Graphs

Principal graphs can be called *data approximators of controllable complexity*. By complexity of the principal objects we mean the following three notions:

- 1) *Geometric complexity*: how far a principal object deviates from its ideal configuration; for the elastic principal graphs we explicitly measure deviation from the 'ideal' pluriharmonic graph by the elastic energy $U_\phi(G)$ (3) (this complexity may be considered as a *measure of non-linearity*);
- 2) *Structural complexity measure*: it is some non-decreasing function of the number of vertices, edges and k -stars of different orders $SC(G) = SC(|V|, |E|, |S_2|, \dots, |S_m|)$; this function penalises for number of structural elements;
- 3) *Construction complexity* is defined with respect to a graph grammar as a number of applications of elementary transformations necessary to construct given G from the simplest graph (one vertex, zero edges).

The construction complexity is defined with respect to a *grammar* of elementary transformation. The graph grammars (Löwe, 1993; Nagl, 1976) provide a well-developed formalism for the description of elementary transformations. An elastic graph grammar is presented as a set of production (or substitution) rules. Each rule has a form $A \rightarrow B$, where A and B are elastic graphs. When this rule is applied to an elastic graph, a copy of A is removed from the graph together with all its incident edges and is replaced with a copy of B with edges that connect B to the graph. For a full description of this language we need the notion of a *labeled graph*. Labels are necessary to provide the proper connection between B and the graph (Nagl, 1976). An approach based on *graph grammars* to constructing effective approximations of an *elastic principal graph* has been proposed recently (Gorban et al, 2007).

Let us define *graph grammar* O as a set of graph grammar operations $O = \{o_1, \dots, o_s\}$. All possible applications of a graph grammar operation o_i to a graph G gives a set of transformations of the initial graph $o_i(G) = \{G_1, G_2, \dots, G_p\}$, where p is the number of all possible applications of o_i to G . Let us also define a sequence of r different graph grammars $\{O^{(1)} = \{o_1^{(1)}, \dots, o_{s_1}^{(1)}\}, \dots, O^{(r)} = \{o_1^{(r)}, \dots, o_{s_r}^{(r)}\}\}$.

Let us choose a grammar of elementary transformations, predefined boundaries of structural complexity SC_{\max} and construction complexity CC_{\max} , and elasticity coefficients λ_i and μ_{kj} .

Definition. *Elastic principal graph* for a dataset X is such an elastic graph G embedded in the Euclidean space by the map $\phi: V \rightarrow \mathbf{R}^m$ that $SC(G) \leq SC_{\max}$, $CC(G) \leq CC_{\max}$, and $U_\phi(G) \rightarrow \min$ over all possible elastic graphs G embeddings in \mathbf{R}^m .

Algorithm for estimating the elastic principal graph

- 1) Initialize the elastic graph G by 2 vertices v_1 and v_2 connected by an edge. The initial map ϕ is chosen in such a way that $\phi(v_1)$ and $\phi(v_2)$ belong to the first principal line in such a way that all the data points are projected onto the principal line segment defined by $\phi(v_1), \phi(v_2)$;
- 2) For all $j=1\dots r$ repeat steps 3-6:
- 3) Apply all grammar operations from $O^{(j)}$ to G in all possible ways; this gives a collection of candidate graph transformations $\{G_1, G_2, \dots\}$;
- 4) Separate $\{G_1, G_2, \dots\}$ into *permissible* and *forbidden* transformations; permissible transformation G_k is such that $SC(G_k) \leq SC_{\max}$, where SC_{\max} is some predefined structural complexity ceiling;
- 5) Optimize the embedding ϕ and calculate the elastic energy $U_\phi(G)$ of graph embedment for every permissible candidate transformation, and choose such a graph G_{opt} that gives the minimal value of the elastic functional: $G_{opt} = \arg \inf_{G_k \in \text{permissible set}} U_\phi(G_k)$;
- 6) Substitute $G \rightarrow G_{opt}$;
- 7) Repeat steps 2-6 until the set of permissible transformations is empty or the number of operations exceeds a predefined number – the construction complexity.

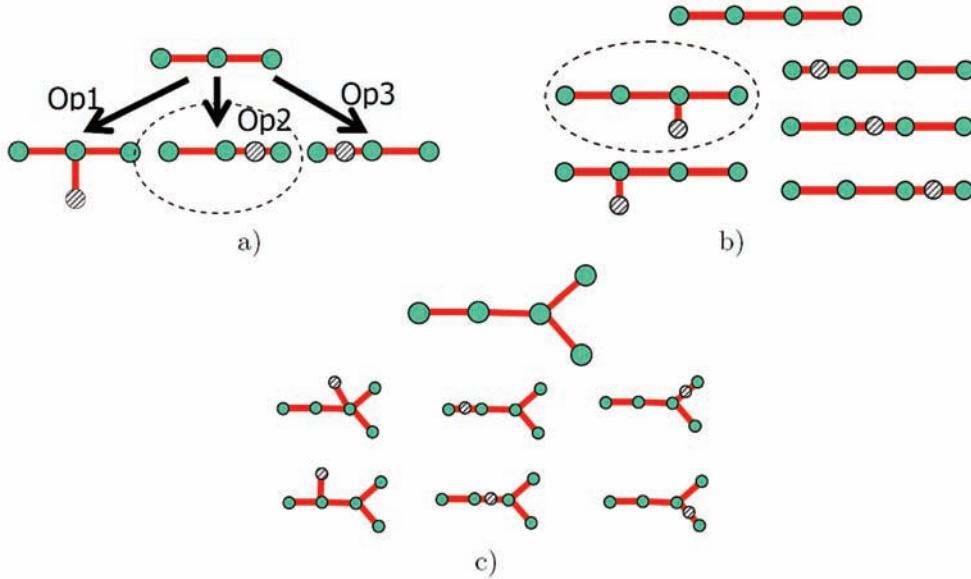
Principal Trees and Metro Maps

Let us construct the simplest non-trivial type of the principal graphs, called principal trees. For this purpose let us introduce a simple ‘Add a node, bisect an edge’ graph grammar (see Fig. 2) applied for the class of primitive elastic graphs.

Definition. *Principal tree* is an acyclic primitive elastic principal graph.

Definition. ‘Remove a leaf, remove an edge’ graph grammar $O^{(\text{shrink})}$ applicable for the class of primitive elastic graphs consists of two operations: 1) The transformation ‘remove a leaf’ can be applied to any vertex v of G with connectivity degree equal to 1: remove v and remove the edge (v, v') connecting v to the tree; 2) The transformation ‘remove an edge’ is applicable to any pair of graph vertices v, v' connected by an edge (v, v') : delete edge (v, v') , delete vertex v' , merge the k -stars for which v and v' are the central nodes and make a new k -star for which v is the central node with a set of neighbors which is the union of the neighbors from the k -stars of v and v' .

Figure 2. Illustration of the simple “add node to a node” or “bisect an edge” graph grammar. a) We start with a simple 2-star from which one can generate three distinct graphs shown. The “Op1” operation is adding a node to a node, operations “Op1” and “Op2” are edge bisections (here they are topologically equivalent to adding a node to a terminal node of the initial 2-star). For illustration let us suppose that the “Op2” operation gives the biggest elastic energy decrement, thus it is the “optimal” operation. b) From the graph obtained one can generate 5 distinct graphs and choose the optimal one. c) The process is continued until a definite number of nodes are inserted.



Definition. ‘Add a node, bisect an edge’ graph grammar $O^{(\text{grow})}$ applicable for the class of primitive elastic graphs consists of two operations: 1) The transformation “add a node” can be applied to any vertex v of G : add a new node z and a new edge (v, z) ; 2) The transformation “bisect an edge” is applicable to any pair of graph vertices v, v' connected by an edge (v, v') : delete edge (v, v') , add a vertex z and two edges, (v, z) and (z, v') . The transformation of the elastic structure (change in the star list) is induced by the change of topology, because the elastic graph is primitive. Consecutive application of the operations from this grammar generates trees, i.e. graphs without cycles.

Also we should define the structural complexity measure $\text{SC}(G)=\text{SC}(|V|, |E|, |S_2|, \dots, |S_m|)$. Its concrete form depends on the application field. Here are some simple examples:

1) $\text{SC}(G)=|V|$: i.e., the graph is considered more complex if it has more vertices;

$$2) \quad \text{SC}(G) = \begin{cases} |S_3|, & \text{if } |S_3| \leq b_{\max} \text{ and } \sum_{k=4}^m |S_k| = 0, \\ \infty, & \text{otherwise} \end{cases}$$

i.e., only b_{\max} simple branches (3-stars) are allowed in the principal tree.

Using the sequence $\{O^{(\text{grow})}, O^{(\text{grow})}, O^{(\text{shrink})}\}$ in the above-described algorithm for estimating the elastic principal graph gives an approximation to the principal trees. Introducing the ‘tree trimming’ grammar

$O^{(shrink)}$ allows to produce principal trees closer to the global optimum, trimming excessive tree branching and fusing k -stars separated by small ‘bridges’.

Principal trees can have applications in data visualization. A principal tree is embedded into a multidimensional data space. It approximates the data so that one can project points from the multidimensional space into the closest node of the tree. The tree by its construction is a one-dimensional object, so this projection performs dimension reduction of the multidimensional data. The question is how to produce a planar tree layout? Of course, there are many ways to layout a tree on a plane without edge intersection. But it would be useful if both local tree properties and global distance relations would be represented using the layout. We can require that

- 1) In a two-dimensional layout, all k -stars should be represented equiangular; this is the small penalty configuration;
- 2) The edge lengths should be proportional to their length in the multidimensional embedding; thus one can represent between-node distances.

This defines a tree layout up to global rotation and scaling and also up to changing the order of leaves in every k -star. We can change this order to eliminate edge intersections, but the result can not be guaranteed. In order to represent the global distance structure, it was found (Gorban et al., 2008) that a good approximation for the order of k -star leaves can be taken from the projection of every k -star on the linear principal plane calculated for all data points, or on the local principal plane in the vicinity of the k -star, calculated only for the points close to this star. The resulting layout can be further optimized using some greedy optimization methods.

The point projections are then represented as pie diagrams, where the size of the diagram reflects the number of points projected into the corresponding tree node. The sectors of the diagram allow us to show proportions of points of different classes projected into the node (see an example on Figure 3).

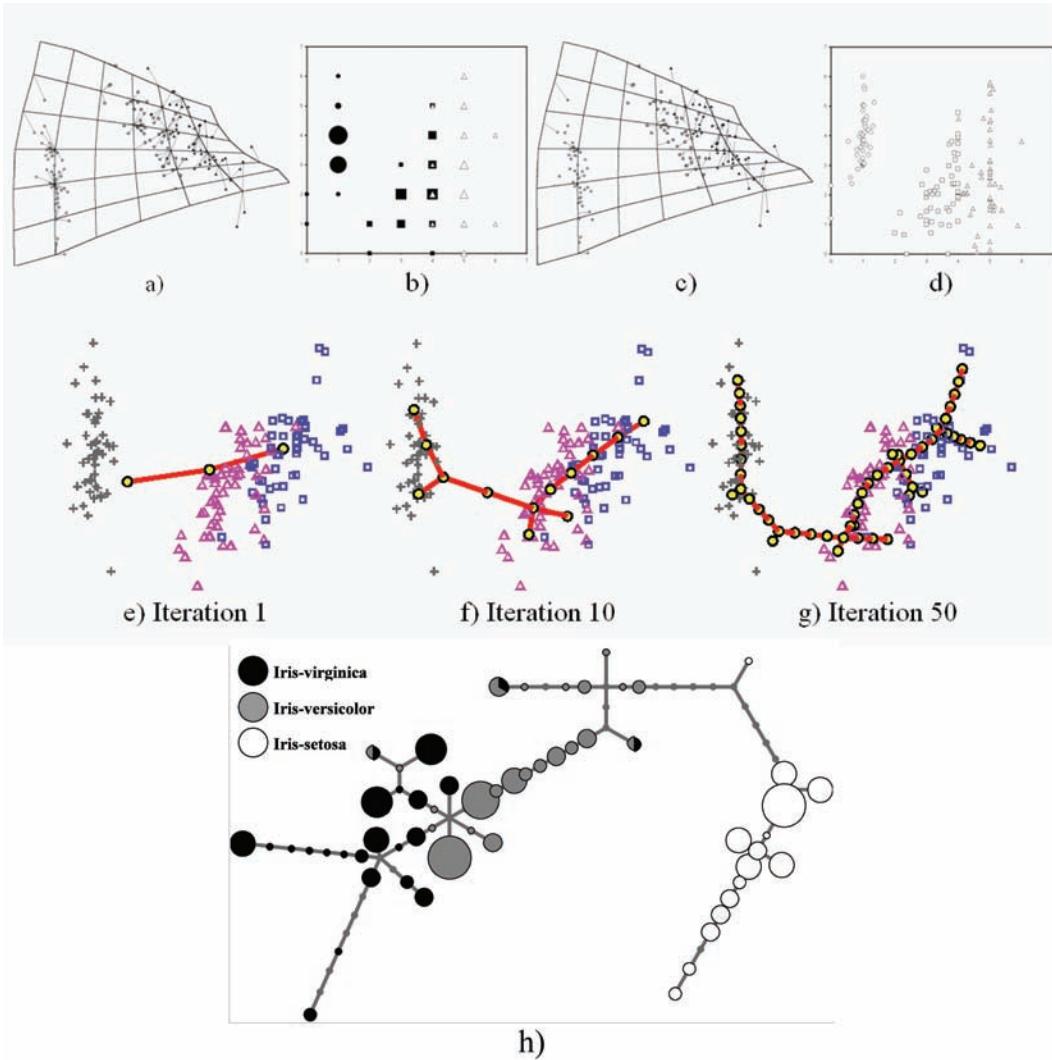
This data display was called a “*metro map*” since it is a schematic and “idealized” representation of the tree and the data distribution with inevitable distortions made to produce a 2D layout. However, using this map one can still estimate the distance from a point (tree node) to a point passing through other points. This map is inherently unrooted (as a real metro map). It is useful to compare this metaphor with trees produced by hierarchical clustering where the metaphor is closer to a “genealogy tree”.

Principal Cubic Complexes

Elastic nets introduced above are characterized by their internal dimension $\dim(G)$. The way to generalize these characteristics on other elastic graphs is to utilize the notion of cubic complex (Gorban et al, 2007).

Definition. *Elastic cubic complex K of internal dimension r* is a Cartesian product $G_1 \times \dots \times G_r$ of elastic graphs G_1, \dots, G_r . It has the vertex set $V_1 \times \dots \times V_r$. Let $1 \leq i \leq r$ and $v_j \in V_j$ ($j \neq i$). For this set of vertices, $\{v_j\}_{j \neq i}$, a copy of G_i in $G_1 \times \dots \times G_r$ is defined with vertices $(v_1, \dots, v_{i-1}, v, v_{i+1}, \dots, v_r)$ ($v \in V_i$), edges $((v_1, \dots, v_{i-1}, v, v_{i+1}, \dots, v_r), (v_1, \dots, v_{i-1}, v', v_{i+1}, \dots, v_r))$, $(v, v') \in E_r$ and, similarly, k -stars of the form $(v_1, \dots, v_{i-1}, S_k, v_{i+1}, \dots, v_r)$, where S_k is a k -star in G_i . For any G_i there are $\prod_{j \neq i} |V_j|$ copies of G_i in G . Sets of

Figure 3. Principal manifold and principal tree for the Iris dataset. a) View of the principal manifold projected on the first two principal components, the data points are shown projected into the closest vertex of the elastic net; b) visualization of data points in the internal coordinates, here classes are represented in the form of Hinton diagrams: the size of the diagram is proportional to the number of points projected, the shape of the diagram denote three different point classes; c) same as a), but the data points are shown projected into the closest point of the piecewise linearly interpolated elastic map; d) same as b), but based on projection shown in c); e)-g) First 50 iterations of the principal tree algorithm, the tree is shown projected onto the principal plane; h) metro map representation of the Iris dataset.



edges and k -stars for Cartesian product are unions of that set through all copies of all factors. A map $\varphi: V_1 \times \dots \times V_r \rightarrow \mathbf{R}^m$ maps all the copies of factors into \mathbf{R}^m too.

Remark. By construction, the energy of the elastic graph product is the energy sum of all factor copies. It is, of course, a quadratic functional of φ .

If we approximate multidimensional data by an r -dimensional object, the number of points (or, more generally, elements) in this object grows with r exponentially. This is an obstacle for grammar-based algorithms even for modest r , because for analysis of the rule $A \rightarrow B$ applications we should investigate all isomorphic copies of A in G . Introduction of a cubic complex is useful factorization of the principal object which allows to avoid this problem.

The only difference between the construction of general elastic graphs and factorized graphs is in the application of the transformations. For factorized graphs, we apply them to factors. This approach significantly reduces the amount of trials in selection of the optimal application. The simple grammar with two rules, “add a node to a node, or bisect an edge,” is also powerful here, it produces products of primitive elastic trees. For such a product, the elastic structure is defined by the topology of the factors.

Incomplete Data

Some of the methods described above allow us to use incomplete data in a natural way. Let us represent an incomplete observation by $\mathbf{x} = (x_1, \dots, @, \dots, @, \dots, x_m)$, where the ‘@’ symbol denotes a missing value.

Definition. *Scalar product between two incomplete observations \mathbf{x} and \mathbf{y} is $(\mathbf{x}, \mathbf{y}) = \sum_{i \neq @}^m x_i y_i$. Then the Euclidean distance is $\sqrt{(\mathbf{x} - \mathbf{y})^2} = \sqrt{\sum_{i \neq @}^m (x_i - y_i)^2}$.*

Remark. This definition has a very natural geometrical interpretation: an incomplete observation with k missing values is represented by a k -dimensional linear manifold L_k , parallel to k coordinate axes corresponding to the missing vector components.

Thus, any method which uses only scalar products or/and Euclidean distances can be applied for incomplete data with some minimal modifications subject to random and not too dense distribution of missing values in X . For example, the iterative method for SVD for incomplete data matrix was developed (Roweis, 1998; Gorban & Rossiev, 1999).

There are, of course, other approaches to incomplete data in unsupervised learning (for example, those presented by Little & Rubin (1987)).

Implicit Methods

Most of the principal objects introduced in this paper are constructed as explicit geometrical objects embedded in \mathbf{R}^m to which we can calculate the distance from any object in X . In this way, they generalize the “data approximation”-based (#1) and the “variation-maximization”-based (#2) definitions of linear PCA. There also exists the whole family of methods, which we only briefly mention here, that generalize the “distance distortion minimization” definition of PCA (#3).

First, some methods take as input a pairwise distance (or, more generally, *dissimilarity*) matrix D and construct such a configuration of points in a low-dimensional Euclidean space that the distance matrix D' in this space reproduce D with maximal precision. The most fundamental in this series is the *metric multidimensional scaling* (Kruskal, 1964). The next is the Kernel PCA approach (Schölkopf et al., 1997) which takes advantage of the fact that for the linear PCA algorithm one needs only the matrix

of pairwise scalar products (Gramm matrix) but not the explicit values of coordinates of X . It allows to apply the kernel trick (Aizerman et al., 1964) and substitute the Gramm matrix by the scalar products calculated with use of some kernel functions. Kernel PCA method is tightly related to the classical multidimensional scaling (Williams, 2002).

Local Linear Embedding or LLE (Roweis & Saul, 2000) searches for such a $N \times N$ matrix A that approximates given \mathbf{x}^i by a linear combination of n vectors-neighbours of \mathbf{x}^i : $\sum_{i=1}^N \|\mathbf{x}^i - \sum_{k=1}^N A_k^i \mathbf{x}^k\|^2 \rightarrow \min$, where only such $A_k^i \neq 0$, if k is one of the n closest to \mathbf{x}^i vectors. After one constructs such a configuration of points in \mathbf{R}^s , $s \ll m$, that $\mathbf{y}^i = \sum_{k=1}^N A_k^i \mathbf{y}^k$, $\mathbf{y}^i \in \mathbf{R}^s$, for all $i = 1 \dots N$. The coordinates of such embedding are given by the eigenvectors of the matrix $(I-A)^T(I-A)$.

ISOMAP (Tenenbaum et al., 2000) and Laplacian eigenmap (Belkin & Niyogi, 2003; Nadler et al., 2008) methods start with construction of the *neighbourhood graph*, i.e. the graph in which close in some sense data points are connected by (weighted) edges. This weighted graph can be represented in the form of a weighted *adjacency matrix* $W = \{W_{ij}\}$. From this graph, ISOMAP constructs a new distance matrix $D^{(ISOMAP)}$, based on the path lengths between two points in the neighbourhood graph, and the multidimensional scaling is applied to $D^{(ISOMAP)}$. The Laplacian map solves the eigenproblem $L\mathbf{f}_\lambda = \lambda S\mathbf{f}_\lambda$, where $S = \text{diag}\{\sum_{j=1}^N W_{0j}, \dots, \sum_{j=1}^N W_{Nj}\}$, $L = S - W$ is the Laplacian matrix. The trivial constant solution corresponding to the smallest eigenvalue $\lambda_0 = 0$ is discarded, while the elements of the eigenvectors $\mathbf{f}_{\lambda_1}, \mathbf{f}_{\lambda_2}, \dots, \mathbf{f}_{\lambda_s}$, where $\lambda_1 < \lambda_2 < \dots < \lambda_s$, give the s -dimensional projection of \mathbf{x}^i , i.e. $P(\mathbf{x}^i) = \{\mathbf{f}_{\lambda_1}(i), \mathbf{f}_{\lambda_2}(i), \dots, \mathbf{f}_{\lambda_s}(i)\}$.

Finally, one can implicitly construct projections into smaller dimensional spaces by training *auto-associative neural networks* with narrow *hidden layer*. An overview of the existing Neural PCA methods can be found in the recent collection of review papers (Gorban et al, 2008).

Example: Principal Objects for the Iris Dataset

On Figure 3 we show application of the elastic principal manifolds and principal trees algorithms to the standard Iris dataset (Fisher, 1936). As expected, two-dimensional approximation of the principal manifold in this case is close to the linear principal plane. One can also see that the principal tree illustrates well the fact of almost complete separation of classes in data space.

Example: Principal Objects for Molecular Surfaces

A molecular surface defines the effective region of space which is occupied by a molecule. For example, the Van-der-Waals molecular surface is formed by surrounding every atom in the molecule by a sphere of radius equal to the characteristic radius of the Van-der-Waals force. After all the interior points are eliminated, this forms a complicated non-smooth surface in 3D. In practice, this surface is sampled by a finite number of points.

Using principal manifolds methodology, we constructed a smooth approximation of such molecular surface for a small piece of a DNA molecule (several nucleotides long). First, we have made an ap-

proximation of this dataset by a 1D principal curve. Interestingly, this curve followed the backbone of the molecule, forming a helix (see Figure 4). Second, we approximated the molecular surface by a 2D manifold. The topology of the surface is expected to be spherical, so we applied spherical topology of the elastic net for optimisation.

We should notice that since it is impossible to make the lengths of all edges equal for the spherical grid, corrections were performed for the edge elasticities during the grid initialization (shorter edges are given larger λ_s). Third, we applied the method for constructing principal cubic complexes, namely, graph product of principal trees, which produced somewhat trivial construction (because no branching was energetically optimal): product of two short elastic principal curves, forming a double helix.

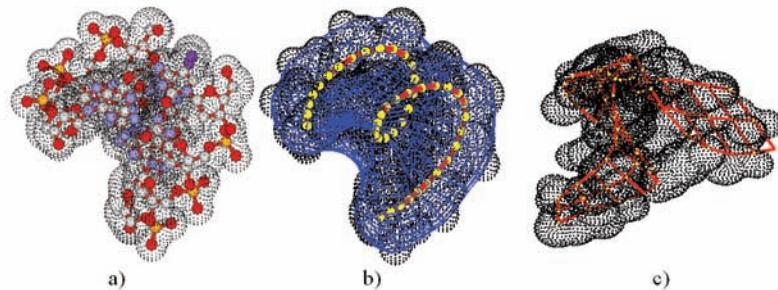
Example: Principal Objects Decipher Genome

A dataset X can be constructed for a string sequence using a short word frequency dictionary approach in the following way: 1) the notion of word is defined; 2) the set of all possible short words is defined, let us say that we have m of them; 3) a number N of text fragments of certain width is sampled from the text; 4) in each fragment the frequency of occurrences of all possible short words is calculated and, thus, each fragment is represented as a vector in multidimensional space \mathbf{R}^m . The whole text then is represented as a dataset of N vectors embedded in \mathbf{R}^m .

We systematically applied this approach to available bacterial genomic sequences (Gorban & Zinovyev, 2008b). In our case we defined: 1) a word is a sequence of three letters from the $\{A,C,G,T\}$ alphabet (triplet); 2) evidently, there are 64 possible triplets in the $\{A,C,G,T\}$ alphabet; 3) we sampled 5000-10000 fragments of width 300 from a genomic sequence; 4) we calculated the frequencies of non-overlapping triplets for every fragment.

The constructed datasets are interesting objects for data-mining, because 1) they have a non-trivial cluster structure which usually contains various configurations of 7 clusters (see Figure 5); 2) class labels can be assigned to points accordingly to available genome annotations; in our case we put information about presence (in one of six possible frameshifts) or absence of the coding information in the current position of a genome; 3) using data mining techniques here has immediate applications in the field of automatic gene recognition and in others, see, for example, (Carbone et al, 2003). On Figure 5 we show

Figure 4. Principal objects approximating molecular surface of a short stretch of DNA molecule. a) stick-and-balls model of the DNA stretch and the initial molecular surface (black points); b) one- and two-dimensional (spherical) principal manifolds for the molecular surface; c) simple principal cubic complex (product of principal trees) which does not have any branching in this case.

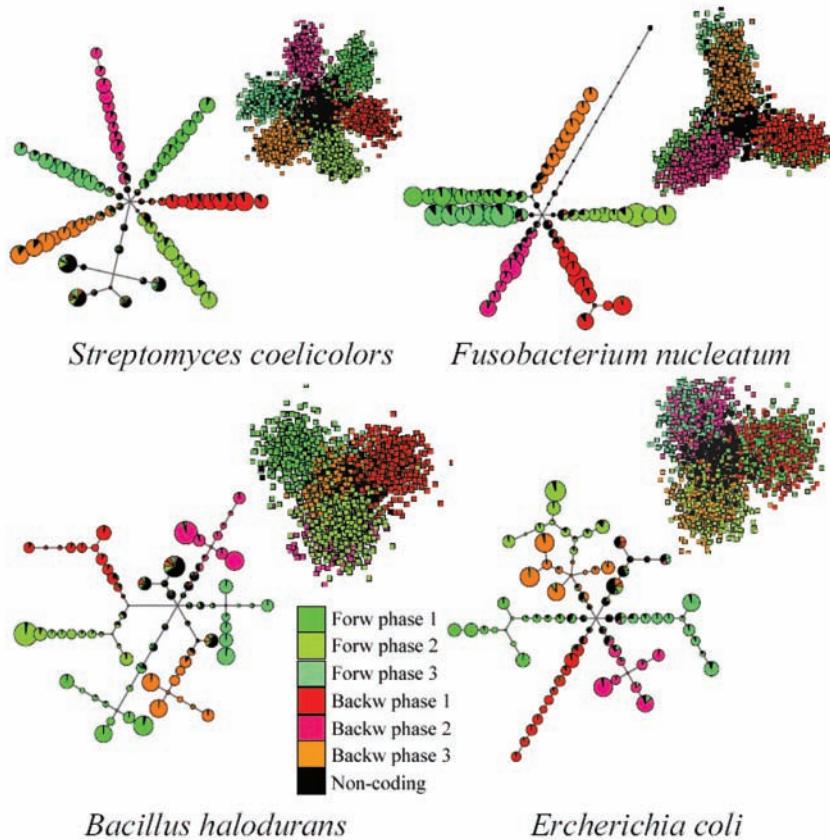


application of both classical PCA and the metro map methods for several bacterial genomes. Look at <http://www.ihes.fr/~zinovyev/7clusters> web-site for further information.

Example: Non-Linear Principal Manifolds for Microarray Data Visualization

DNA microarray data is a rich source of information for molecular biology (an expository overview is provided by Leung & Cavalieri (2003)). This technology found numerous applications in understanding various biological processes including cancer. It allows to screen simultaneously the expression of all genes in a cell exposed to some specific conditions (for example, stress, cancer, treatment, normal conditions). Obtaining a sufficient number of observations (chips), one can construct a table of “samples

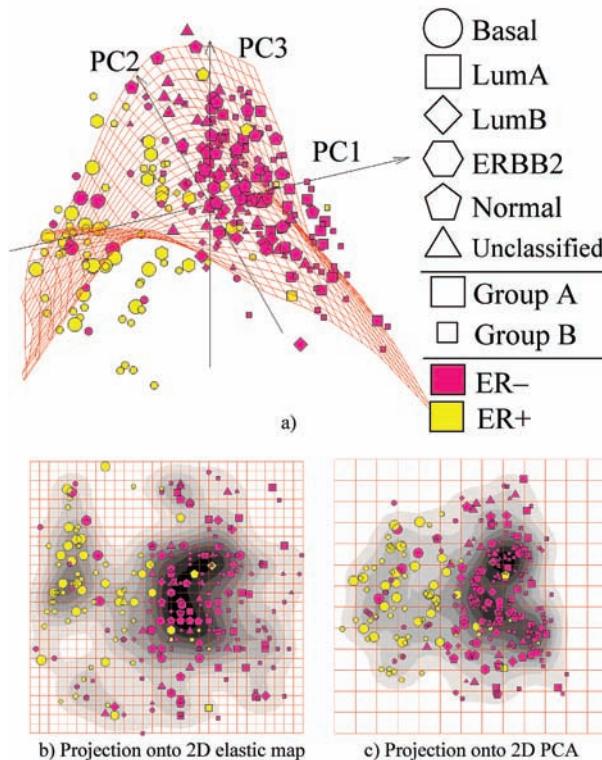
*Figure 5. Seven cluster structures presented for 4 selected genomes. A genome is represented as a collection of points (text fragments represented by their triplet frequencies) in the 64-multidimensional space. Color codes denote point classes corresponding to 6 possible frameshifts when a random fragment overlaps with a coding gene (3 in the forward and 3 in the backward direction of the gene), and the black color corresponds to non-coding regions. For every genome a principal tree (“metro map” layout) is shown together with 2D PCA projection of the data distribution. Note that the clusters that appear to be mixed on the PCA plot for *Escherichia coli* (they remain mixed in 3D PCA as well) are well separated on the “metro map”. This proves that they are well-separated in \mathbf{R}^{64} .*



vs genes”, containing logarithms of the expression levels of, typically several thousands (n) of genes, in typically several tens (m) of samples.

On Figure 6 we provide a comparison of data visualization scatters after projection of the breast cancer dataset, provided by Wang et al. (2003), onto the linear two- and non-linear two-dimensional principal manifold. The latter one is constructed by the elastic maps approach. Each point here represents a patient treated from cancer. Before dimension reduction it is represented as a vector in \mathbf{R}^n , containing the expression values for all n genes in the tumor sample. Linear and non-linear 2D principal manifolds provide mappings $\mathbf{R}^n \rightarrow \mathbf{R}^2$, drastically reducing vector dimensions and allowing data visualization. The form, the shape and the size of the point on the Fig.6 represent various clinical data (class labels) extracted from the patient’s disease records.

Figure 6. Visualization of breast cancer microarray dataset using elastic maps. Ab initio classifications are shown using points size (ER, estrogen receptor status), shape (Group A – patients with aggressive cancer, Group B – patients with non-aggressive cancer) and color (TYPE, molecular type of breast cancer). a) Configuration of nodes projected into the three-dimensional principal linear manifold. One clear feature is that the dataset is curved such that it can not be mapped adequately onto a two-dimensional principal plane. b) The distribution of points in the internal non-linear manifold coordinates is shown together with estimation of the two-dimensional density of points. c) The same as b) but for the linear two-dimensional manifold. One can notice that the “basal” breast cancer subtype is much better separated on the non-linear mapping and some features of the distribution become better resolved.



Practical experience from bioinformatics studies shows that two-dimensional data visualization using non-linear projections allow to catch more signals from data (in the form of clusters or specific regions of higher point density) than linear projections, see Figure 6 and a good example by Ivakhno & Armstrong (2008).

In addition to that, Gorban & Zinovyev (2008a) performed a systematic comparison of performance of low-dimensional linear and non-linear principal manifolds for microarray data visualization, using the following four criteria: 1) mean-square distance error; 2) distortions in mapping the big distances between points; 3) local point neighbourhood preservation; 4) compactness of point class labels after projection. It was demonstrated that non-linear two-dimensional principal manifolds provide systematically better results accordingly to all these criteria, achieving the performance of three- and four- dimensional linear principal manifolds (principal components).

The interactive ViMiDa (Visualization of Microarray Data) and ViDaExpert software allowing microarray data visualization with use of non-linear principal manifolds are available on the web-site of Institut Curie (Paris): <http://bioinfo.curie.fr/projects/vidaexpert> and <http://bioinfo.curie.fr/projects/vimida>.

CONCLUSION

In this chapter we gave a brief practical introduction into the methods of construction of principal objects, i.e. objects embedded in the ‘middle’ of the multidimensional data set. As a basis, we took the unifying framework of mean squared distance approximation of finite datasets which allowed us to look at the principal graphs and manifolds as generalizations of the mean point notion.

REFERENCES

- Aizerman, M., Braverman, E., & Rozonoer, L. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25, 821–837.
- Aluja-Banet, T., & Nonell-Torrent, R. (1991). Local principal component analysis. *Qüestio*, 3, 267-278.
- Arthur, D., & Vassilvitskii, S. (2007). K-means++ the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007*, New Orleans, LA, USA (pp. 1027-1035).
- Bau, D., III, & Trefethen, L. N. (1997). *Numerical linear algebra*. Philadelphia, PA: SIAM Press.
- Bauer, F. L. (1957). Das verfahren der treppeniteration und verwandte verfahren zur losung algebraischer eigenwert probleme. *Math. Phys.*, 8, 214–235.
- Belkin, M., & Niyogi, P. (2006). Laplacian Eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6), 1373–1396. doi:10.1162/089976603321780317
- Bishop, C. M., Svensén, M., & Williams, C. K. I. (1998). GTM: The generative topographic mapping. *Neural Computation*, 10(1), 215–234. doi:10.1162/089976698300017953

- Braverman, E. M. (1970). Methods of extremal grouping of parameters and problem of apportionment of essential factors. *Automation and Remote Control*, (1), 108–116.
- Cochran, R. N., & Horne, F. H. (1977). Statistically weighted principal component analysis of rapid scanning wavelength kinetics experiments. *Analytical Chemistry*, 49, 846–853. doi:10.1021/ac50014a045
- Delicado, P. (2001). Another look at principal curves and surfaces. *Journal of Multivariate Analysis*, 77, 84–116. doi:10.1006/jmva.2000.1917
- Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B. Methodological*, 39(1), 1–38.
- Diday, E. (1979). *Optimisation en classification automatique, Tome 1,2* (in French). Rocquencourt, France: INRIA.
- Durbin, R., & Willshaw, D. (1987). An analogue approach to the travelling salesman problem using an elastic net method. *Nature*, 326, 689–691. doi:10.1038/326689a0
- Efron, B. (1967). The two sample problem with censored data. In *Proc. of the Fifth Berkeley Simp. Math. Statist. Probab.* 4 (pp. 831–853). Berkeley, CA: Univ.California Press.
- Einbeck, J., Evers, L., & Bailer-Jones, C. (2008). Representing complex data using localized principal components with application to astronomical data. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 178–201). Berlin, Germany: Springer.
- Einbeck, J., Tutz, G., & Evers, L. (2005). Local principal curves. *Statistics and Computing*, 15, 301–313. doi:10.1007/s11222-005-4073-8
- Erwin, E., Obermayer, K., & Schulten, K. (1992). Self-organizing maps: Ordering, convergence properties and energy functions. *Biological Cybernetics*, 67, 47–55. doi:10.1007/BF00201801
- Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7, 179–188.
- Flury, B. (1990). Principal points. *Biometrika*, 77, 33–41. doi:10.1093/biomet/77.1.33
- Fréchet, M. (1948). Les éléments aléatoires de nature quelconque dans un espace distancié. *Ann. Inst. H. Poincaré*, 10, 215–310.
- Fukunaga, K., & Olsen, D. (1971). An algorithm for finding intrinsic dimensionality of data. *IEEE Transactions on Computers*, 20(2), 176–183. doi:10.1109/T-C.1971.223208
- Gabriel, K. R., & Zamir, S. (1979). Lower rank approximation of matrices by least squares with any choices of weights. *Technometrics*, 21, 298–489. doi:10.2307/1268288
- Gorban, A., Kégl, B., Wunsch, D., & Zinovyev, A. (Eds.). (2008). *Principal manifolds for data visualization and dimension reduction*. Berlin, Germany: Springer.
- Gorban, A., Sumner, N., & Zinovyev, A. (2007). Topological grammars for data approximation. *Applied Mathematics Letters*, 20(4), 382–386. doi:10.1016/j.aml.2006.04.022

- Gorban, A., Sumner, N. R., & Zinovyev, A. (2008). Beyond the concept of manifolds: Principal trees, metro maps, and elastic cubic complexes. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 219-237). Berlin, Germany: Springer.
- Gorban, A., & Zinovyev, A. (2005). Elastic principal graphs and manifolds and their practical applications. *Computing*, 75, 359–379. doi:10.1007/s00607-005-0122-6
- Gorban, A., & Zinovyev, A. (2008a). Elastic maps and nets for approximating principal manifolds and their application to microarray data visualization. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 96-130). Berlin, Germany: Springer.
- Gorban, A., & Zinovyev, A. (2008b). PCA and K-means decipher genome. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 309-323). Berlin, Germnay: Springer.
- Gorban, A. N., Pitenko, A. A., Zinov'ev, A. Y., & Wunsch, D. C. (2001). Vizualization of any data using elastic map method. *Smart Engineering System Design*, 11, 363–368.
- Gorban, A. N., & Rossiev, A. A. (1999). Neural network iterative method of principal curves for data with gaps. *Journal of Computer and Systems Sciences International*, 38(5), 825–830.
- Gorban, A. N., Zinovyev, A. Y., & Wunsch, D. C. (2003). Application of the method of elastic maps in analysis of genetic texts. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN*, Portland, Oregon.
- Hartigan, J. A. (1975). *Clustering algorithms*. New York: Wiley.
- Hastie, T. (1984). *Principal curves and surfaces*. Unpublished doctoral dissertation, Stanford University, CA.
- Hotelling, H. (1933). Analisys of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24, 417–441. doi:10.1037/h0071325
- Ivakhno, S., & Armstrong, J. D. (2007). Non-linear dimensionality reduction of signalling networks. *BMC Systems Biology*, 1(27).
- Jolliffe, I. T. (2002). *Principal component analysis, series: Springer series in statistics, 2nd ed., XXIX*. New York: Springer.
- Kambhatla, N., & Leen, T. K. (1997). Dimension reduction by local PCA. *Neural Computation*, 9, 1493–1516. doi:10.1162/neco.1997.9.7.1493
- Karhunen, K. (1946). Zur spektraltheorie stochastischer prozesse. *Suomalainen Tiedeakatemia Toimituksia. Sar. A.4: Biologica*, 37.
- Kégl, B. (1999). *Principal curves: Learning, design, and applications*. Unpublished doctoral dissertation, Concordia University, Canada.

- Kégl, B., & Krzyzak, A. (2002). Piecewise linear skeletonization using principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1), 59–74. doi:10.1109/34.982884
- Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43, 59–69. doi:10.1007/BF00337288
- Kohonen, T. (1997). *Self-organizing maps*. Berlin, Germany: Springer.
- Koren, Y., & Carmel, L. (2004). Robust linear dimensionality reduction. *IEEE Transactions on Visualization and Computer Graphics*, 10(4), 459–470. doi:10.1109/TVCG.2004.17
- Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29, 1–27. doi:10.1007/BF02289565
- Leung, Y. F., & Cavalieri, D. (2003). Fundamentals of cDNA microarray data analysis. *Trends in Genetics*, 19(11), 649–659. doi:10.1016/j.tig.2003.09.015
- Little, R. J. A., & Rubin, D. B. (1987). *Statistical analysis with missing data*. New York: John Wiley.
- Liu, Z.-Y., Chiu, K.-C., & Xu, L. (2003). Improved system for object detection and star/galaxy classification via local subspace analysis. *Neural Networks*, 16, 437–451. doi:10.1016/S0893-6080(03)00015-7
- Lloyd, S. (1957). *Least square quantization in PCM's* (Bell Telephone Laboratories Paper).
- Loève, M. M. (1955). *Probability theory*. Princeton, NJ: VanNostrand.
- Löwe, M. (1993). Algebraic approach to single-pushout graph transformation. *Theoretical Computer Science*, 109, 181–224. doi:10.1016/0304-3975(93)90068-5
- Lumley, J. L. (1967). The structure of inhomogeneous turbulent flows. In A. M. Yaglom & V. I. Tatarski (Eds.), *Atmospheric turbulence and radio propagation* (pp. 166-178). Moscow, Russia: Nauka.
- MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proc. of the 5th Berkeley Symp. on Math. Statistics and Probability* (pp. 281-297).
- Mirkin, B. (2005). *Clustering for data mining: A data recovery approach*. Boca Raton, FL: Chapman and Hall.
- Mulier, F., & Cherkassky, V. (1995). Self-organization as an iterative kernel smoothing process. *Neural Computation*, 7, 1165–1177. doi:10.1162/neco.1995.7.6.1165
- Nadler, B., Lafon, S., Coifman, R., & Kevrekidis, I. G. (2008). Diffusion maps - a probabilistic interpretation for spectral embedding and clustering algorithms. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 238-260). Berlin, Germany: Springer.
- Nagl, M. (1976). Formal languages of labelled graphs. *Computing*, 16, 113–137. doi:10.1007/BF02241984
- Ostrovsky, R., Rabani, Y., Schulman, L. J., & Swamy, C. (2006). The effectiveness of Lloyd-type methods for the k-means problem. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science* (pp. 165-176). Washington, DC: IEEE Computer Society.

- Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 6(2), 559–572.
- Pelleg, D., & Moore, A. (1999). Accelerating exact k -means algorithms with geometric reasoning. In *Proceedings of the Fifth International Conference on Knowledge Discovery in Databases* (pp. 277-281). Menlo Park, CA: AAAI Press.
- Ritter, H., Martinetz, T., & Schulten, K. (1992). *Neural Computation and Self-Organizing maps: An introduction*. Reading, MA: Addison-Wesley.
- Roweis, S. (1998). EM algorithms for PCA and SPCA. In *Advances in neural information processing systems* (pp. 626-632). Cambridge, MA: MIT Press.
- Roweis, S., & Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290, 2323–2326. doi:10.1126/science.290.5500.2323
- Schölkopf, B., Smola, A. J., & Müller, K.-R. (1997). Kernel principal component analysis. In *Proceedings of the ICANN* (pp. 583-588).
- Smola, A. J., Mika, S., Schölkopf, B., & Williamson, R. C. (2001). Regularized principal manifolds. *Journal of Machine Learning Research*, 1, 179–209. doi:10.1162/15324430152748227
- Steinhaus, H. (1956). Sur la division des corps materiels en parties. In *Bull. Acad. Polon. Sci., Cl. III vol IV* (pp. 801-804).
- Strang, G. (1993). The fundamental theorem of linear algebra. *The American Mathematical Monthly*, 100(9), 848–855. doi:10.2307/2324660
- Sylvester, J. J. (1889). On the reduction of a bilinear quantic of the n th order to the form of a sum of n products by a double orthogonal substitution. *Messenger of Mathematics*, 19, 42–46.
- Tarpey, T., & Flury, B. (1996). Self-consistency: A fundamental concept in statistics. *Statistical Science*, 11(3), 229–243. doi:10.1214/ss/1032280215
- Tenenbaum, J. B., de Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290, 2319–2323. doi:10.1126/science.290.5500.2319
- Tibshirani, R. (1992). Principal curves revisited. *Statistics and Computing*, 2, 183–190. doi:10.1007/BF01889678
- Verbeek, J. J., Vlassis, N., & Kröse, B. (2002). A k -segments algorithm for finding principal curves. *Pattern Recognition Letters*, 23(8), 1009–1017. doi:10.1016/S0167-8655(02)00032-6
- Wang, Y., Klijn, J. G., & Zhang, Y. (2005). Gene expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365, 67–679.
- Williams, C. K. I. (2002). On a connection between kernel PCA and metric multidimensional scaling. *Machine Learning*, 46, 11–19. doi:10.1023/A:1012485807823
- Xu, R., & Wunsch, D. (2008). *Clustering*. New York: IEEE Press / John Wiley & Sons.

Yin, H. (2008). Learning nonlinear principal manifolds by self-organising maps. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 68-95). Berlin, Germany: Springer.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, 67*(2), 301–320. doi:10.1111/j.1467-9868.2005.00503.x

KEY TERMS AND DEFINITIONS

Principal Components: Such an orthonormal basis in which the covariance matrix is diagonal.

Principal Manifold: Intuitively, a smooth manifold going through the middle of data cloud; formally, there exist several definitions for the case of data distributions: 1) Hastie and Staelze's principal manifolds are self-consistent curves and surfaces; 2) Kegl's principal curves provide the minimal mean squared error given the limited curve length; 3) Tibshirani's principal curves maximize the likelihood of the additive noise data model; 4) Gorban and Zinovyev elastic principal manifolds minimize a mean square error functional regularized by addition of energy of manifold stretching and bending; 5) Smola's regularized principal manifolds minimize some form of a regularized quantization error functional; and some other definitions.

Principal Graph: A graph embedded in the multidimensional data space, providing the minimal mean squared distance to the dataset combined with deviation from an “ideal” configuration (for example, from pluriharmonic graph) and not exceeding some limits on complexity (in terms of the number of structural elements and the number of graph grammar transformations needed for obtaining the principal graph from some minimal graph).

Self-Consistent Approximation: Approximation of a dataset by a set of vectors such that every point \mathbf{y} in the vector set is a conditional mean of all points from dataset that are projected in \mathbf{y} .

Expectation/Maximisation Algorithm: Generic splitting algorithmic scheme with use of which almost all algorithms for estimating principal objects are constructed; it consists of two basic steps: 1) projection step, at which the data is projected onto the approximator, and 2) maximization step, at which the approximator is optimized given the projections obtained at the previous step.

Chapter 3

Learning Algorithms for RBF Functions and Subspace Based Functions

Lei Xu

Chinese University of Hong Kong and Beijing University, PR China

ABSTRACT

Among extensive studies on radial basis function (RBF), one stream consists of those on normalized RBF (NRBF) and extensions. Within a probability theoretic framework, NRBF networks relates to nonparametric studies for decades in the statistics literature, and then proceeds in the machine learning studies with further advances not only to mixture-of-experts and alternatives but also to subspace based functions (SBF) and temporal extensions. These studies are linked to theoretical results adopted from studies of nonparametric statistics, and further to a general statistical learning framework called Bayesian Ying Yang harmony learning, with a unified perspective that summarizes maximum likelihood (ML) learning with the EM algorithm, RPCL learning, and BYY learning with automatic model selection, as well as their extensions for temporal modeling. This chapter outlines these advances, with a unified elaboration of their corresponding algorithms, and a discussion on possible trends.

BACKGROUND

The renaissance of neural network and then machine learning since the 1980's is featured by two streams of extensive studies, one on multilayer perceptron and the other on radial basis function networks or shortly RBF net. While a multilayer perceptron partitions data space by hyperplanes and then making subsequent processing via nonlinear transform, a RBF net partitions data space into modular regions via local structures, called radial or non-radial basis functions. After extensive studies on multilayer perceptron, studies have been turned to RBF net since the late 1980's and the early 1990's, with a wide range applications (Kardirkamanathan, Niranjan, & Fallside, 1991; Chang & Yang, 1997; Lee, 1999;

DOI: 10.4018/978-1-60566-766-9.ch003

Mai-Duy & Tran-Cong, 2001; Er, 2002; Reddy & Ganguli, 2003; Lin & Chen 2004; Isaksson, Wisell, & Ronnow, 2005; Sarimveis, Doganis, & Alexandridis, 2006; Guerra & Coelho, 2008; Karami & Mommadi, 2008).

In the literature of machine learning, advances on RBF net can be roughly divided into two streams. One stems from the literature of mathematics on multivariate function interpolation and spline approximation, as well as Tikhonov type regularization for ill-posed problems, which were brought to neural networks learning by (Powell, 1987; Broomhead & Lowe, 1998; Poggio & Girosi, 1990; Yuille & Grzywacz, 1989) and others. Actually, it is shown that RBF net can be naturally derived from Tikhonov regularization theory, i.e. least square fitting subjected to a rotational and translational constraint term by a different stabilizing operator (Poggio & Girosi, 1990; Yuille & Grzywacz, 1989). Moreover, RBF net has also been shown to have not only the universal approximation ability possessed by multilayer perceptron (Hartman, 1989; Park & Sandberg, 1993), but also the best approximation ability (Poggio & Girosi, 1990) and a good generalization property (Botros & Atkeson, 1991).

The other stream can be traced back to Parzen Window estimator proposed in the early 1960's. Studies of this stream progress via active interactions between the literature of statistics and the literature of machine learning. During 1970's-80's, Parzen window has been widely used for estimating probability density, especially for estimating the class densities that are used for Bayesian decision based classification in the literature of pattern recognition. Also, it has been introduced into the literature of neural networks under the name of probabilistic neural net (Specht, 1990). By that time, it has not yet been related to RBF net since it does not directly relates to the above discussed function approximation purpose.

In the literature of neural networks and machine learning, Moody & Darken (1989) made a popular work via Gaussian bases as follows:

$$f(x) = \sum_{j=1}^k w_j \varphi_j(x - c_j, \Sigma_j), \quad \varphi_j(x - c_j, \Sigma_j) = e^{-0.5(x - c_j)^T \Sigma_j^{-1} (x - c_j)} / \sum_{j=1}^k e^{-0.5(x - c_j)^T \Sigma_j^{-1} (x - c_j)}. \quad (1)$$

This normalized type of RBF net is featured by

$$\sum_{j=1}^k \varphi_j(x - c_j, \Sigma_j) = 1, \quad (2)$$

and thus usually called Normalized RBF (RBF). Moody & Darken (1989) actually considered a special case $\Sigma_j = \sigma_j^2 I$. Unknowns are learned from a set of samples via a two stage implementation. First, a clustering algorithm (e.g., k-means) is used to estimate c_i as each cluster's center. Second, $\varphi_j(x)$ is calculated for every sample and unknowns w_j are estimated by a least square linear fitting. Nowlan (1990) proceeded along this line with Σ_j considered in its general form such that not only the receptive field of each base function can be elliptic instead of radial symmetrical, but also the well known EM algorithms (Redner & Walker, 1984) are adopted for estimating c_i and Σ_j with better performances than that by a clustering algorithm.

In the nonparametric statistics literature, extensive studies have also been made towards regression tasks under the name of kernel regression estimator, which is an extension of Parzen window estimator from density estimation to statistical regression (Devroye, 1981&87). In (Xu, Krzyzak, & Yuille,

1992&94), kernel regression estimators are shown to be special cases of NRBF net. In help of this connection, theoretical results available on kernel regression have been adopted, which cast new lights on the theoretical analysis of NRBF net in several aspects, e.g., universal approximation extended to statistical consistency, convergence extended to convergence rate, etc.

Another line of studies in the early 1990 is featured by using the mixtures-of-experts (ME) (Jacobs, Jordan, Nowlan, & Hinton, 1991) for regression:

$$f(x) = \sum_{j=1}^k g_j(x, \phi) f_j(x, \theta_j), \quad (3)$$

where each $v_j(x, \phi)$ is a function implemented by a three layer forward networks, and $0 \leq g_j(x, \phi) \leq 1$ is called gating net. This $f(x) = \sum_{j=1}^k g_j(x, \phi) f_j(x, \theta_j)$ is actually a regression equation of a mixture of conditional distribution as follows:

$$q(z | x) = \sum_{j=1}^k g_j(x, \phi) q(z | x, j), \text{ with } f(x) = E_{q(z|x)} z \text{ and } f_j(x, \theta_j) = E_{q(z|x,j)} z. \quad (4)$$

One typical example is $q(z | x, j) = G(z | f_j(x, \theta_j), \Gamma_j)$, where $G(u | \mu, \Sigma)$ denotes a Gaussian density with mean vector μ and covariance matrix Σ . All the unknowns in this mixture are estimated via the maximum likelihood (ML) learning by a generalized EM algorithm (Jordan & Jacobs, 1994; Jordan & Xu, 1995).

Furthermore, a connection has been built between this ME model and NRBF net such that the EM algorithm has been adopted for estimating jointly all the parameters in a NRBF net to improve suboptimal performances due to a two stage implementation. In sequel, we start at this EM algorithm, and then introduce further advances on NRBF studies. Not only basis functions is extended from radial to subspace based functions, and further to temporal modeling, but also studies further proceed to tackle model selection problem, i.e., how to determine the number k and the subspace dimensions.

THE EM ALGORITHM AND BEYOND: ALTERNATIVE ME VERSUS ENRBF

Xu, Jordan & Hinton (1994&95) proposed an alternative mixtures-of-experts (AME) via a different gating net. Considering each expert $G(z | f_j(x, \theta_j), \Gamma_j)$ supported on a Gaussian $G(x | \mu_j, \Sigma_j)$ or generally an expert $q(z | x, \theta_j)$ supported on a corresponding $q(x | \phi_j)$, we have that $p(z | x)$ is supported on a finite mixture $q(x | \phi)$ with its corresponding posteriori as the gating net:

$$q(z | x) = \sum_{j=1}^k g_j(x, \phi) q(z | x, \theta_j), \quad g_j(x, \phi) = \frac{\alpha_j q(x | \phi_j)}{q(x | \phi)}, \quad q(x | \phi) = \sum_{j=1}^k \alpha_j q(x | \phi_j), \quad 0 \leq \alpha_j, \quad \sum_{j=1}^k \alpha_j = 1. \quad (5)$$

Figure 1. Algorithm 1: EM algorithm for alternative mixture of experts and NRBF nets

E step : getting $p(j z_t, x_t) = \frac{\alpha_j q(x_t \phi_j)q(z_t x_t, \theta_j)}{\sum_{\ell=1}^k \alpha_\ell q(x_t \phi_\ell)q(z_t x_t, \theta_\ell)}$;
M step : fixing $p(j z_t, x_t)$ and updating the unknowns $\{\alpha_j, \phi_j, \theta_j\}$ to max/incr $\sum_{t=1}^N p(j z_t, x_t) \ln [\alpha_j q(x_t \phi_j)q(z_t x_t, \theta_j)]$ which is further decoupled into <ul style="list-style-type: none"> • update ϕ_j to max/incr $\sum_{t=1}^N p(j z_t, x_t) \ln q(x_t \phi_j)$, • update θ_j to max/incr $\sum_{t=1}^N p(j z_t, x_t) \ln q(z_t x_t, \theta_j)$, • get $\alpha_j = \frac{1}{N} \sum_{t=1}^N p(j z_t, x_t)$. * max/incr Φ means 'maximize or increase Φ '.
(a)
A Special Case : NRBF nets $q(x \phi_j) = G(x \mu_j, \Sigma_j)$ subject to $\alpha_j / \Sigma_j ^{0.5} \propto \text{const}$ thus $g_j(x, \phi) = \frac{\alpha_j q(x \phi_j)}{q(x \phi)} = \frac{\exp[-0.5(x - c_j)^T \Sigma_j^{-1} (x - c_j)]}{\sum_{\ell=1}^k \exp[-0.5(x - c_\ell)^T \Sigma_\ell^{-1} (x - c_\ell)]} = \phi_j(x - c_j, \Sigma_j)$ <p>also, α_j in M step is replaced by $\alpha_j = \frac{ \Sigma_j ^{0.5}}{\sum_\ell \Sigma_\ell ^{0.5}}$.</p>
(b)

Given a training set $\{z_t, x_t\}_{t=1}^N$, parameter learning is made by the ML learning on the joint distribution $q(z | x)q(x | \phi) = \sum_{j=1}^k \alpha_j q(x | \phi_j)q(z | x, \theta_j)$, which is implemented by the EM algorithm, i.e., Algorithm 1(a). Details are referred to (Xu, Jordan & Hinton, 1995).

We observe that the M step decouples the updating in parts. The one for updating θ_j of each expert is basically same as in (Jacobs, Jordan, Nowlan, & Hinton, 1991) except here we get a different $p(j | z_t, x_t)$ as the E step. The ones for updating the unknowns α_j, ϕ_j of the gate can be made via letting $p(j | z_t, x_t)$ to take the role of $p(j | x_t)$ in the M step of the EM algorithm on a finite mixture or Gaussian mixture (Redner & Walker, 1984; Nowlan, 1990).

Alternatively, we can use the resulted $p(j | z_t, x_t)$ to replace $g_j(x, \phi)$ as the gate in eqn.(5), i.e.,

$$q(z_t | x_t) = \sum_{\ell=1}^k p(\ell | z_t, x_t)q(z_t | x_t, \theta_\ell). \quad (6a)$$

When z_t is discrete, this equation is still directly computable on testing samples. However, when z_t takes real values that are usually not available on testing samples, we can not directly use $p(j | z_t, x_t)$ to

replace $g_j(x, \phi)$ as the gate in eqn.(4). Instead, we get an estimate $f_j(x_t, \theta_j) = E_{q(z|x,j)} z$ as \hat{z}_t , from which we usually have either $q(\hat{z}_t | x_t, \theta_j)$ in a constant or $q(\hat{z}_t | x_t, \theta_j) = \varphi(\theta_j)$, e.g., $\varphi(\theta_j) = (2\pi)^{-0.5d} |\Gamma_j|^{-0.5}$ for $q(z | x, \theta_j) = G(z | f_j(x, \theta_j), \Gamma_j)$. Thus, we get

$$\begin{aligned} p(j | x, z) &= \frac{\alpha_j q(x | \phi_j) q(z | x, \theta_j)}{\sum_{j=1}^k \alpha_j q(x | \phi_j) q(z | x, \theta_j)}, \quad \text{for an integer } z, \\ p(j | x, \hat{z}_t) &= \frac{\alpha_j q(x | \phi_j) \varphi(\theta_j)}{\sum_{j=1}^k \alpha_j q(x | \phi_j) \varphi(\theta_j)}, \quad \text{for a real } z. \end{aligned} \quad (6b)$$

which takes the place of $g_j(x, \phi)$ as the gate in eqn.(4).

Furthermore, NRBF net is obtained as a special case, as shown in Algorithm I(b). In this case, the regression function in eqn. (3) becomes

$$f(x) = \sum_{j=1}^k f_j(x, \theta_j) \varphi_j(x - c_j, \Sigma_j), \quad (7)$$

which is actually an extension of NRBF net with a constant w_j generalized to a general regression $f_j(x, \theta_j)$. That is, we are lead back to eqn.(2) when $f_j(x, \theta_j) = w_j$ and the so called Extended NRBF (ENRBF) net (Xu, Jordan & Hinton, 1994&95; Xu, 1998) when $f_j(x, \theta_j) = W_j x + w_j$. Straightforwardly, we can use the EM algorithm in Algorithm I to update all the unknowns, from which we can also obtain different versions of the EM algorithms at various particular cases (Xu, 1998). For an example, it improves the two stage implementation of learning NRBF by (Nowlan, 1990) in a sense that the updating of c_i , Σ_j also takes w_j in consideration via $p(j | z_r, x_r)$, while the updating of w_j takes c_i and Σ_j in consideration via this $p(j | z_r, x_r)$ too.

It is interesting to further elaborate the connection between RBF net and the ME models from a dual perspective. For the RBF net by eqn.(2), we seek a combination of a series of basis functions $\varphi_j(x - c_j, \Sigma_j)$ via linear weights of w_j . For the ME model by eqn.(3), we seek a convex combination of a series of functions $f_j(x, \theta_j)$ weighted by $g_j(x, \phi)$ that actually takes the role of the basis function $\varphi_j(x - c_j, \Sigma_j)$ in eqn.(2). That is, there is a dual perspective that can swap the roles of the weighting coefficients and what are weighted. The ME perspective provides extensions of RBF net from classical radial basis functions $\varphi_j(x - c_j, \Sigma_j)$ to $g_j(x, \phi) = \alpha_j G(x | \mu_j, \Sigma_j) / \sum_{j=1}^k \alpha_j G(x | \mu_j, \Sigma_j)$ and further to $p(j | z_r, x_r)$ in eqn.(6a) and eqn.(6b). Moreover, in addition to NRBF net with $f_j(x, \theta_j) = w_j$ and ENRBF net with $f_j(x, \theta_j) = W_j x + w_j$, $f_j(x, \theta_j)$ can also be implemented by a multilayer networks (Jacobs, Jordan, Nowlan, & Hinton, 1991), which actually combines the features of both NRBF net and three layer forward networks.

TWO STAGE IMPLEMENTATION VERSUS AUTOMATIC MODEL SELECTION

Using a structure by a RBF net or a ME model to accommodate a mapping $x \rightarrow z$, one needs a learning algorithm that estimates unknown parameters in this structure. The performance of a resulted RBF or ME can be measured via a set of testing samples by either a square error $\|z_t - f(x_t)\|^2$ for function approximation or classification error for pattern recognition, as well as the likelihood $\ln p(z_t | x_t)$ for distribution approximation. Correspondingly, an algorithm is guided by one of these purposes via a set of training set. The classic RBF net studies considers minimizing the square error $\|z_t - f(x_t)\|^2$ on a set $\mathbf{X}_N = \{x_t\}_{t=1}^N$ of training samples. The studies (Jacobs, Jordan, Nowlan, & Hinton, 1991; Jordan & Jacobs, 1995) on the ME model considers maximizing the likelihood $\ln p(z_t | x_t)$ on \mathbf{X}_N , while the alternative ME considers maximizing the likelihood $\ln p(z_t, x_t)$ on \mathbf{X}_N (Xu, Jordan & Hinton, 1994&95). However, a good performance on a training set is not necessarily good on a testing set, especially when the training set consists of a small size of samples. The reason is too many free parameters to be determined. Studies towards this problem have been widely studied along two directions in the literature of statistics and machine learning.

One is called regularization that adds some constraint or regularity to the unknown parameters, the selected structure, and the training samples (Powell, 1987; Poggio, & Girosi, 1990). Readers are referred to (Xu, 2007c) for a summary of typical regularization techniques studied in the literature of learning. One example is smoothing \mathbf{X}_N by a Gaussian kernel with a zero mean and a unknown smoothing parameter h as its variance, i.e., instead of directly using \mathbf{X}_N for training we consider $p(\mathbf{X} | \mathbf{X}_N, h) = G(\mathbf{X} - \mathbf{X}_N | 0, h^2 I)$.

Actually, maximizing $\ln p(z_t, x_t)$ by the alternative ME can be regarded as maximizing $\ln p(z_t | x_t)$ plus a term $\ln q(x_t | \phi)$ that adds certain constrain on samples of x_t . This is an example of a family of typical regularization techniques, featured by the format $\ln p(z_t | x_t)$ or $\ln p(z_t | x_t) + \lambda \Omega(\theta, z_t, x_t)$, where $\Omega(\theta)$ or $\Omega(\theta, z_t, x_t)$ is usually called stabilizer or regularizor, and λ is called regularization strength.

However, we encounter two difficulties to impose a regularization. One is the difficulty of appropriately controlling a regularization strength λ , which is usually able to be roughly estimated only for a rather simple structure via either handling the integral of marginal density or in help of cross validation (Stone, 1978), but with very extensive computing costs. The other is the difficulty of choosing an appropriate $\Omega(\theta)$ or $\Omega(\theta, x_t)$ based on a priori knowledge that we are usually not available and difficult to get. Instead, an isotropic or nonspecific stabilizer is usually imposed, e.g., $\Omega(\theta) = \|\theta\|^2$. This type of isotropic or nonspecific regularization faces a dilemma. We need a regularization on a structure $\mathbf{S}_k(\Theta_k)$ with extra parts, while an isotropic or nonspecific regularization can not discard the extra parts but still let them in action to blur those useful parts. For an example, given a set of samples $\{z_t, x_t\}$ that come from a curve $z = x^2 + 3x + 2$, if we use a polynomial $z = \sum_{i=0}^k a_i x^i$ of an order $k > 2$ to fit the data set, we desire to force all the parameters $\{a_i, i \geq 3\}$ to be zero, while minimizing $\|\theta\|^2 = \sum_{i=0}^k a_i^2$ fails to treat the parameters $\{a_i, i \geq 3\}$ differently from the parameters $\{a_i, i \leq 2\}$.

To tackle the problems, we turn to consider the other direction that consists of those efforts made under the name of model selection. It refers to select an appropriate one among a family of infinite many candidate structures $\{\mathbf{S}_k(\Theta_k)\}$ with each $\mathbf{S}_k(\Theta_k)$ in a same configuration but in different scales, each of which is labeled by a scale parameter k in term of one integer or a set of integers. Selecting an

appropriate k means getting a structure that consists of an appropriate number of free parameters. For a structure by a RBF net or a ME model, this k is simply the number k of bases or experts. Usually, a maximum likelihood (ML) principle based learning is not good for model selection. The EM algorithm in Algorithm I works well with a pre-specified number k of bases or experts. The performance will be affected by whether an appropriate k is selected, while how to determine k is a critical problem.

Many efforts have been made towards model selection for over three decades in past. The classical one is making a two stage implementation. First, enumerate a candidate set \mathbf{K} of k and estimate the unknown set Θ_k of parameters by ML learning for a solution Θ_k^* at each $\mathbf{k} \in \mathbf{K}$. Second, use a model selection criterion $J(\Theta_k^*)$ to select a best \mathbf{k}^* . Several criteria are available for the purpose, such as AIC, CAIC, BIC, cross validation, etc (Stone, 1978; Akaike, 1981; Bozdogan, 1987; Wallace& Freeman, 1987; Cavanova, 1997; Rissanen, 1989; Vapnik, 1995&2006). Also, readers are referred to (Xu, 2007c) for a recent elaboration and comparison. Some of these criteria (e.g., AIC, BIC) have also been adopted for selecting the number k of basis functions in RBF networks (Konishi, Ando, & Imoto, 2004). Unfortunately, anyone of these criteria usually provides a rough estimate that may not yield a satisfactory performance. Even with a criterion $J(\Theta_k)$ available, this two stage approach usually incurs a huge computing cost. Still, the parameter learning performance deteriorates rapidly as k increases, which makes the value of $J(\Theta_k)$ evaluated unreliably.

One direction that tackles this challenge is featured by incremental algorithms that attempts to incorporate as much as possible what learned as k increases step by step. Its focus is on learning newly added parameters, e.g., the studies made on mixture of factor analyses (Ghahramani & Beal, 2000; Salah and & Alpaydin, 2004). Such an incremental implementation can save computing costs. However, it usually leads to suboptimal performance because not only those newly added parameters but also the old parameter set Θ_k actually have to be re-learned. This suboptimal problem is lessened by a decremental implementation that starts with k at a large value and reduces k step by step. At each step, one takes a subset out of Θ_k with the remaining parameter updated, and discard the subset with a biggest decreasing of $J(\Theta_k^*)$ after trying a number of such subsets. Such a procedure can be formulated as a tree searching. The initial parameter set Θ_k is the root of the tree, and discarding one subset moves to one immediate descendent. A depth-first searching suffers from a suboptimal performance seriously, while a breadth-first searching suffers a huge combinatorial computing cost. Usually, a trade off between the two extremes is adopted.

One other direction of studies is called automatic model selection. An early effort of this direction is Rival Penalized Competitive Learning (RPCL) (Xu, Krzyzak, & Oja, 1992&93) for adaptively learning the centers c_i and Σ_j of radial basis in NRBF networks (Xu, Krzyzak & Oja, 1992&93; Xu, 1998a), with the number k automatically determined during learning. The key idea is that not only the winner c_w moves a little bit to adapt the current sample x_t , but also the rival (i.e., the second winner) c_r is repelled a little bit from x_t to reduce a duplicated information allocation. As a result, an extra c_j will be driven far away from data with its corresponding $\alpha_j \rightarrow 0$ and $Tr[\Sigma_j] \rightarrow 0$. Moreover, RPCL learning algorithm has been proposed for jointly learning not only c_i and Σ_j but also $W_j x + w_j$ (Sec.4.4 & Table 2, Xu, 2001&02). In general, RPCL is applicable to any $S_k(\Theta_k)$ that consists of k individual substructures. With k initially at a value larger enough, a coming sample x_t is allocated to one of the k substructures via competition, and the winner adapts this sample by a little bit, while the rival is de-learned a little bit to reduce a duplicated allocation. This rival penalized mechanism will discard those extra substructures, making model selection automatically during learning. Various extensions have been made in the

past decades (Xu,2007d). Instead of the heuristic mechanism embedded in RPCL algorithm, Bayesian Ying-Yang (BYY) harmony learning was proposed in (Xu,1995) and systematically developed in the past decade, which leads us to not only new model selection criteria but also algorithms with automatic model selection ability via maximizing a Ying Yang harmony functional.

In general, this automatic model selection direction demonstrates a quite difference nature from the usual incremental or decremental implementation that bases on evaluating the change $J(\Theta_k) - J(\Theta_k \cup \theta_\Delta)$ as a subset θ_Δ of parameters is added or removed. Thus, automatic model selection is associated with a learning algorithm or a learning principle with the following two features:

- There is an indicator $\rho(\theta_r)$ on a subset $\theta_r \in \Theta_k$, we have $\rho(\theta_r) = 0$ if θ_r consists of parameters of a redundant structural part.
- In implementation of this learning algorithm or optimizing this learning principle, there is an mechanism that automatically drives $\rho(\theta_r) \rightarrow 0$ θ_r towards a specific value. Thus, the corresponding redundant structural part is effectively discarded.

Shown in Algorithm 2 (displayed in Figure 2) is a summary of three types of adaptive algorithms for alternative ME, including NRBF and ENRBF as special cases. The implementation is almost the same for three types of learning, except taking different values of $\eta_{j,t}$. Considering k individual substructures, each has a local measure $H_t(\Theta_j)$ on its fitness to a coming sample x_t and the parameters Θ_j are updated by updating rules with same format to three types of learning. The updating rules are obtained from the gradient $\nabla_{\Theta_j} H_t(\Theta_j)$ either directly or in a modified direction that has a positive projection on this gradient, as explained in Figure 3(E). According to Figure 3(A), each updating direction is modified by $\eta_{j,t}$ on its direction and step size.

We get a further insight at a special case that $h_x = 0$, $h_z = 0$ (thus Step (D) discarded). The first allocating scheme is $\eta_{j,t} = p_{j,t} = p_t(j | \Theta) = p(j | z_t, x_t)$, i.e., same as the E step in Algorithm I. Actually, Algorithm II at this setting is an adaptive EM algorithm for ML learning (Xu, 1998). The second scheme only takes values at the winner and the rival, while the negative value of $\eta_{j,t} = p_{j,t} = -\gamma$ reverses the updating direction such that the rival is de-learned a little bit to reduce its fitness $H_t(\Theta_r)$ to the current sample x_t . It extends the RPCL learning for NRBF from a two stage implementation (Xu, Krzyzak, &Oja, 1992&93) to an improved implementation that updates all the unknowns jointly (Xu, 1998). One problem to the RPCL learning is how to choose an appropriate $\gamma > 0$. The third scheme $\eta_{j,t} = p_{j,t}(1 + \delta h_{t,j})$ shares the features of both the EM learning and RPCL learning without needing a pre-specified $\gamma > 0$, which is derived from the BYY harmony learning to be introduced in the sequel.

BYY HARMONY LEARNING: FUNDAMENTALS

As shown in the left –top corner of Figure 4, a set $\mathbf{X} = \{x\}$ of samples are regarded as generated via a top-down path from its inner representation $\mathbf{R} = \{\mathbf{Y}, \Theta\}$, with a long term memory Θ that is a collection of all unknown parameters in the system for collectively representing the underlying structure of \mathbf{X} , and with a short term memory \mathbf{Y} that each element $y \in \mathbf{Y}$ is the corresponding inner representation of one element $x \in X$. A mapping $\mathbf{R} \rightarrow \mathbf{X}$ and an inverse $\mathbf{X} \rightarrow \mathbf{R}$ are jointly considered via the joint

Figure 2. Algorithm 2: Adaptive learning algorithm for alternative ME and NRBF nets

$H_t(\Theta_j) = \pi_t(\Theta_j) - \frac{1}{2} \text{Tr}[h_x^2(\Pi_j^x + \Pi_j^{z|x}) + h_z^2\Pi_j^z] + \frac{1}{N} \ln \{q(h_x)q(h_z)\}, \Pi_j^x = -\nabla_x^2 \ln q(x|\varphi_j), \pi_t(\Theta_j) = \ln[\alpha_j q(x_t|\varphi_j)q(z_t|x_t,\theta_j)], \Pi_j^{z|x} = -\nabla_x^2 \ln q(z|x,\theta_j), \Pi_j^z = -\nabla_z^2 \ln q(z|x,\theta_j)$, where $\nabla_x^2 f(x) = \partial^2 \ln f(x)/\partial x \partial x^T$, $\text{Tr}[A]$ denotes the trace of a matrix A

(a)

We maximize $\sum_{j=1}^k p_j(j|\Theta)H_t(\Theta_j)$ by gradient based rules in Tab.1(E) via

$$dH_0(p \parallel q, \Theta, \mathbf{k}, \Xi) = \sum_{j=1}^k p_{j,t}(1+\delta h_{t,j})d\pi(x_t, \Theta_j),$$

$$\delta h_{t,j} = \pi(x_t, \Theta_j) - \sum_{\ell=1}^k p_{\ell,t}\pi(x_t, \Theta_\ell), p_t(j|\Theta) = \exp[\pi_t(\Theta_j)]/\sum_{\ell=1}^k \exp[\pi_t(\Theta_\ell)].$$

(b)

This maximization is made by iterating the following two steps until converged :

YING STEP Choose one allocating scheme in Tab.1(A) and

get $p_{j,t}, \delta h_{j,t}$ based on the current value of Θ .

YANG STEP Let $\eta_{j,t} = p_{j,t}(1+\delta h_{j,t})$, update :

$$(a) \alpha_j^{new} = e^{\mathbf{c}_j^{old} + \Delta \mathbf{c}_j} / \sum_{\ell} e^{\mathbf{c}_{\ell}^{old} + \Delta \mathbf{c}_{\ell}}, \mathbf{c} = [c_1, \dots, c_k]^T, \mathbf{a} = [\alpha_1, \dots, \alpha_k]^T, \mathbf{1} = [1, \dots, 1]^T,$$

$$\Delta \mathbf{c} \propto (I - \mathbf{a} \mathbf{1}^T) \mathbf{g}_a, \mathbf{g}_a = \text{diag}[p_{1,t}(1+\delta h_{1,t}), \dots, p_{k,t}(1+\delta h_{k,t})],$$

$$(\because \sum_{j=1}^k p_{j,t}(1+\delta h_{j,t}) d \ln \alpha_j = \text{Tr}[\mathbf{g}_a^T d \mathbf{a}] = \text{Tr}[\mathbf{g}_a^T (I - \mathbf{a} \mathbf{1}^T) d \mathbf{c}])$$

$$\text{or simply } \alpha_j = |\Sigma_j|^{0.5} / \sum_{\ell} |\Sigma_{\ell}|^{0.5} \text{ for NRBF or ENRBF.}$$

If $\alpha_j \rightarrow 0$, discard the corresponding structure and its Θ_j .

$$(b) \varphi_j + \Delta \varphi_j \in D_{\varphi_j} \text{ with } \Delta \varphi_j \propto \eta_{j,t} \nabla_{\varphi_j \in D_{\varphi_j}} \ln q(x_t|\varphi_j);$$

$$(c) \theta_j + \Delta \theta_j \in D_{\theta_j} \text{ with } \Delta \theta_j \propto \eta_{j,t} \nabla_{\theta_j \in D_{\theta_j}} \ln q(z_t|x_t, \theta_j),$$

where $u + \Delta u \in D_u$ means 'updating within the domain D_u of u' .

$$(d) h_z^{new} = h_z^{old} + \Delta h_z \text{ with } \Delta h_z \propto \{\frac{1}{N} q'(h_z) - h_z \sum_{j=1}^k p_{j,t} \text{Tr}[\Pi_j^z]\}, f'(r) = \frac{df(r)}{dr},$$

$$h_x^{new} = h_x^{old} + \Delta h_x \text{ with } \Delta h_x \propto \{\frac{1}{N} q'(h_x) - h_x \sum_{j=1}^k p_{j,t} \text{Tr}[\Pi_j^x + \Pi_j^{z|x}]\}.$$

(e)

distribution of \mathbf{X}, \mathbf{R} in two types of Bayesian decomposition. In a compliment to the famous ancient Ying-Yang philosophy, the decomposition of $p(\mathbf{X}, \mathbf{R})$ coincides the Yang concept with a visible domain $p(\mathbf{X})$ for a Yang space and a forward pathway by $p(\mathbf{R} | \mathbf{X})$ as a Yang pathway. Thus, $p(\mathbf{X}, \mathbf{R})$ is called Yang machine. Also, $q(\mathbf{X}, \mathbf{R})$ is called Ying machine with an invisible domain $q(\mathbf{R})$ for a Ying space and a backward pathway by $q(\mathbf{X} | \mathbf{R})$ as a Ying pathway. Such a Ying-Yang pair is called Bayesian Ying-Yang (BYY) system. It further consists of two layers. The front layer is itself a Ying-Yang pair for $\mathbf{X} \rightarrow \mathbf{Y}$ and $\mathbf{Y} \rightarrow \mathbf{X}$. The back layer supports the front layer by a priori $q(\Theta | \Xi)$, while $p(\mathbf{R} | \mathbf{X})$ consists of the posteriori $p(\Theta | \mathbf{X}, \Xi)$ that transfers the knowledge from observations to the back layer.

The input to the Ying Yang system is through $p(\mathbf{X} | \Theta_x) = p(\mathbf{X} | \mathbf{X}_N, h)$ obtained from a sample set $\mathbf{X}_N = \{x_t\}_{t=1}^N$, e.g., $p(\mathbf{X} | \mathbf{X}_N, h) = G(\mathbf{X} - \mathbf{X}_N | 0, h^2 I)$. To build up an entire system, we need to

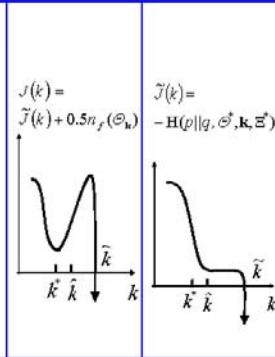
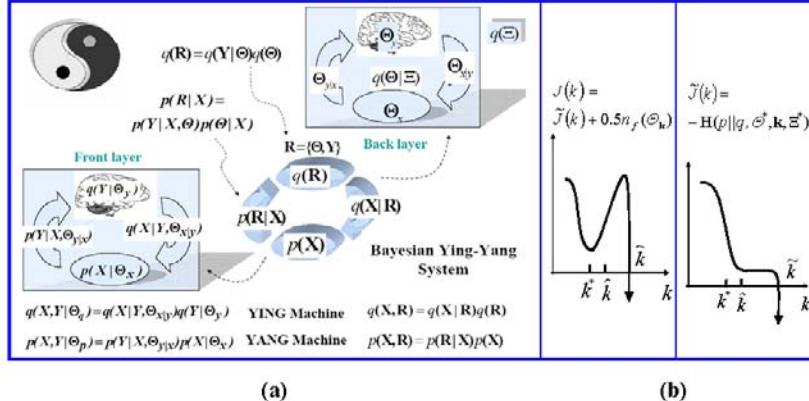
Figure 3. Allocating schemes, typical examples, and gradient based updating

EM	RPLC	BYY
$\delta h_{i,j} = 0$	$\delta h_{i,j} = 0$	$\delta h_{i,j} = \pi(x_i, \Theta_j) - \sum_{\ell=1}^k p_\ell(j \Theta) \pi(x_i, \Theta_j)$
$p_{j,t} = p_t(j \Theta)$	$p_{j,t} = \bar{\delta}_{j,\ell^*} - \gamma \bar{\delta}_{j,r^*}$	$p_{j,t} = p_t(j \Theta)$
$p_t(j \Theta) = \frac{e^{\pi_t(\Theta_j)}}{\sum_{\ell=1}^k e^{\pi_t(\Theta_\ell)}}, \quad \bar{\delta}_{i,\ell} = \begin{cases} 1, & i = \ell, \quad \ell^* = \operatorname{argmax}_j \pi(x_i, \Theta_j); \\ 0, & i \neq \ell, \quad r^* = \operatorname{argmax}_{j \neq \ell} \pi(x_i, \Theta_j). \end{cases}$ <p style="text-align: center;">(A)</p>		
$q(x \varphi_j) = G(x \mu_j, \Sigma_j), \quad \Pi_j^x = \Sigma_j^{-1}, \quad \eta_{j,t} = p_{j,t}(1 + \delta h_{i,j}), \quad \Delta \mu_j \propto \eta_{j,t} e_{j,t}, \quad e_{j,t} = x_t - \mu_j,$ $\Sigma_j = S_j S_j^T, \quad G_{\Sigma_j} = \Sigma_j^{-1} (e_{j,t} e_{j,t}^T + 0.5 h_z^2 I - \Sigma_j) \Sigma_j^{-1}, \quad \Delta \Sigma_j \propto \eta_{j,t} (e_{j,t} e_{j,t}^T + 0.5 h_z^2 I - \Sigma_j) \Sigma_j^{-T}$ <p style="text-align: center;">(B)</p>		
$q(z x, \theta_j) = G(z W_j^{z x} x + w_j^{z x}, \Sigma_j^{z x}), \quad \Pi_j^z = \Sigma_j^{z x-1}, \quad \Pi_j^{z x} = W_j^{z x} \Sigma_j^{z x-1} W_j^{z x}$ $\Delta w_j^{z x} \propto \eta_{j,t} \varepsilon_{j,t}, \quad \varepsilon_{j,t} = z_t - (W_j^{z x} x_t + w_j^{z x}), \quad \Delta W_j^{z x} \propto \eta_{j,t} \varepsilon_{j,t} x_t^T - h_z^2 W_j^{z x}, \quad \Sigma_j^{z x} = \Gamma_j \Gamma_j^T,$ $G_{\Sigma_j^{z x}} = \Sigma_j^{z x-1} (\varepsilon_{j,t} \varepsilon_{j,t}^T + h_z^2 I - \Sigma_j^{z x}) \Sigma_j^{z x-1},$ $\Delta \Gamma_j \propto \eta_{j,t} \Sigma_j^{z x} G_{\Sigma_j^{z x}} \Gamma_j = \eta_{j,t} (\varepsilon_{j,t} \varepsilon_{j,t}^T + h_z^2 I - \Sigma_j^{z x}) \Gamma_j^{-T}.$ <p style="text-align: center;">(C)</p>		
$q(z x, \theta_j) = q(z = \ell x, \theta_j) = e^{\tilde{z}_j^{(\ell)}} / \sum_\ell e^{\tilde{z}_j^{(\ell)}} = q_j^{(\ell)}, \quad \tilde{z}_j = [\tilde{z}_j^{(1)}, \dots, \tilde{z}_j^{(k)}]^T$ $\mathbf{q}_j = [q_j^{(1)}, \dots, q_j^{(k)}]^T, \quad \tilde{z}_j = W_j^{z x} x + w_j^{z x} - \frac{1}{2} [x^T V_{j,k}^{z x} x, \dots, x^T V_{j,k}^{z x} x]^T$ $\Delta w_j^{z x} \propto \eta_{j,t} \varepsilon_{j,t}, \quad \varepsilon_{j,t} = [\tilde{\delta}_{1,\ell}, \dots, \tilde{\delta}_{k,\ell}]^T - \mathbf{q}_{j,t}, \quad q_{j,t}^{(\ell)} = q(z_t = \ell x_t, \theta_j)$ $\Delta V_{j,t}^{z x} \propto -\eta_{j,t} \varepsilon_{j,t} x_t^T, \quad \Delta W_j^{z x} \propto \eta_{j,t} \{ \varepsilon_{j,t} x_t^T - h_z^2 (\operatorname{diag}[\mathbf{q}_{j,t}] - \mathbf{q}_{j,t} \mathbf{q}_{j,t}^T) W_j^{z x} \}$ <p style="text-align: center;">(D)</p>		
$\operatorname{Max/incr } f(\theta, \Sigma) \text{ with a vector } \theta \text{ and a positive definite matrix } \Sigma \text{ via}$ $\mathbf{g}_\theta = \nabla_\theta f, \quad \mathbf{G}_\Sigma = \nabla_\Sigma f. \quad \text{We update } \theta^{\text{new}} = \theta^{\text{old}} + \Delta \theta \text{ with}$ <ul style="list-style-type: none"> • Gradient updating: $\Delta \theta \propto \mathbf{g}_\theta$ which means $\Delta \theta = \gamma \mathbf{g}_\theta$ with a small scalar γ. • Projected Gradient: $\Delta \theta \propto P \mathbf{g}_\theta$ by a positive definite P, with $\mathbf{g}_\theta^T P \mathbf{g}_\theta > 0$. update $\Sigma = S \Sigma^T$ via $S^{\text{new}} = S^{\text{old}} + \Delta S$ to keep Σ positive definite. • Gradient updating: $\Delta S \propto \mathbf{G}_S \mathbf{G}_S = \mathbf{G}_\Sigma \mathbf{S}$, since $\operatorname{Tr}[G_x^T d\Sigma] = 2 \operatorname{Tr}[(G_\Sigma \mathbf{S})^T dS]$. • Projected Gradient: $\Delta S \propto P \mathbf{G}_S$ or $\Delta S \propto \mathbf{G}_S P$ by a positive definite P. ($\because \operatorname{Tr}[\mathbf{G}_S^T P \mathbf{G}_S] = \operatorname{vec}(\mathbf{G}_S)^T (I \otimes P) \operatorname{vec}(\mathbf{G}_S) > 0$) • Bi-projected Gradient: $\Delta S \propto P \mathbf{G}_S P$ by a positive definite P. ($\because \operatorname{Tr}[\mathbf{G}_S^T P \mathbf{G}_S P] = \operatorname{vec}(\mathbf{G}_S)^T (P \otimes P) \operatorname{vec}(\mathbf{G}_S) > 0$) <p>Remarks: (a) $\operatorname{vec}(A)$ stacks all the columns of A into a vector, and \otimes is Kronecker product. (b) P is chosen to simplify computation, and may speed up convergence too.</p> <p style="text-align: center;">(E)</p>		

design appropriate structures for the rests in the system. Different designs perform different learning tasks and thus typical learning models are unified under a general framework. As shown in Figure 4, designs are guided by the following three principles, respectively:

- **Least Redundancy Principle** Subject to the nature of learning tasks, $q(\mathbf{R})$ should be in a structure for the inner representation of \mathbf{X}_N encoded with a redundancy as least as possible. First, the number of variables and parameters of $\mathbf{R} = \{\mathbf{Y}, \Theta\}$ should be as less as possible, which is itself the model selection task that is beyond the task of design. Second, the dependences among these

Figure 4. Bayesian Ying-Yang system and best harmony learning



Distributive Measure		Collective Measure (entropy)	
Likelihood based	$U(p) \begin{cases} \propto \ln p(u) + c, & (\text{a}) \\ = \ln p(u), & (\text{b}) \end{cases}$	Renyi	$U(p) = \frac{1}{1-\alpha} \log \int p^\alpha(u) du$
Hessian based	$U(p) \begin{cases} \propto \nabla_{uu^T}^2 \ln p(u), & (\text{a}) \\ = \nabla_{uu^T}^2 \ln p(u), & (\text{b}) \end{cases}$	Shannon	$U(p) = - \int p(u) \ln p(u) du$

where “ \propto ” means proportional, and $\nabla_{uu^T}^2 f(u) = \partial^2 f(u) / \partial u \partial u^T$ is applicable to a twice differentiable $f(u)$. Both the notations will be used subsequently.

Remarks:

- “ \propto ” is not meaningful to a collective measure since it is just a real scalar but provides a relaxation on a distributive measure.
- $U(p)=U(q)$ means $p=aq^b$ with $\int p=1$ for Likelihood (a), and we get $a=1, b=1$ for Likelihood (b).
- In a Hessian measure, $U(p)=U(q)$ at point u^* means that p, q get a local uncertainty conversation in a neighbor of u^* in a sense of equal local convexity.
- Likelihood based implies Hessian based if p, q are twice differentiable. Also, Renyi implies Shannon, and Renyi reduces to Shannon for $\alpha=1$.

(c)

variables should be as least as possible, which is possibly or at least partly to be considered. E.g., when \mathbf{Y} consists of multiple components $\mathbf{Y} = \{Y^{(j)}\}$, we design $q(\mathbf{Y} | \Theta) = \prod_j q(Y^{(j)} | \theta_y^{(j)})$.

- **Divide and Conquer Principle** Subject to the representation formats of \mathbf{X} and \mathbf{Y} , a complicated mapping $\mathbf{Y} \rightarrow \mathbf{X}$ is modeled by designing $q(\mathbf{X} | \mathbf{R}) = q(\mathbf{X} | \mathbf{Y}, \Theta)$ via a mixture of a number of simple structures. E.g., we may design $q(\mathbf{X} | \mathbf{Y}, \Theta)$ via a mixture of a number of linear regression featured by Gaussians of \mathbf{Y} conditional on \mathbf{X} . In some situation, it is not necessary to design $q(\mathbf{R})$ and $q(\mathbf{X} | \mathbf{R})$ separately. Instead, we design $q(\mathbf{X}, \mathbf{R})$ (especially $q(\mathbf{X}, \mathbf{Y}, \Theta_q)$) via a single parametric model as a whole but still attempting to follow the above two principles. One example is a product Gaussian mixture, as the one encountered in Type B of Rival penalized competitive learning (see Xu, 2007b). In general, we may consider an integrable measure $M(\mathbf{X}, \mathbf{Y}, \Theta_q)$ by

$$q(\mathbf{X}, \mathbf{Y}, \Theta_q) = M(\mathbf{X}, \mathbf{Y}, \Theta_q) / \int M(\mathbf{X}, \mathbf{Y}, \Theta_q) d\mathbf{X} d\mathbf{Y}$$

- **Uncertainty Conversation Principle** In a compliment to the Ying-Yang philosophy, Ying is primary and thus is designed first, while Yang is relatively secondary and thus is designed basing on Ying. Moreover, as illustrated by the Ying-Yang sign located at the top-left of Figure 4, the room of varying or dynamic range of Yang should balance with that of Ying, which motivates to design $p(\mathbf{R}, \mathbf{R})$ (in fact, only $p(\mathbf{R} | \mathbf{X})$ because we have $p(\mathbf{X} | \Theta_x) = p(\mathbf{X} | \mathbf{X}_N, h)$ already) under a principle of uncertainty conversation between Ying-Yang. In other words, Yang machine preserves a varying room or dynamic range that is appropriate to accommodate uncertainty or information contained within the Ying machine. That is $U(p(\mathbf{X}, \mathbf{R})) = U(q(\mathbf{X}, \mathbf{R}))$ under a uncertainty measure $U(p)$, in one of the choices within the table in Figure 4. Since a Yang machine consists of two components $p(\mathbf{R} | \mathbf{X})p(\mathbf{X})$, we may consider one or both of the uncertainty conversations as follows:

$$\begin{aligned} U(p(\mathbf{R} | \mathbf{X})) &= U(q(\mathbf{R} | \mathbf{X})), \quad q(\mathbf{R} | \mathbf{X}) = q(\mathbf{X}, \mathbf{R}) / q(\mathbf{X}) \text{ and} \\ U(p(\mathbf{X})) &= U(q(\mathbf{X})), \quad q(\mathbf{X}) = \int q(\mathbf{X}, \mathbf{R}) d\mathbf{R} \end{aligned}$$

The above uncertainty conservation may occur at three different levels. One is likelihood based conservation that the structure of Yang machine gets a strong link to the ones of Ying machine. Considering $\sum_L p(L | \mathbf{X}, \theta_{L|X}) = 1$, the Yang structure is actually given by the Bayesian inverse of Ying machine $p(L | \mathbf{X}, \theta_{L|X}) = q(\mathbf{X}, \mathbf{R}) / \sum_L q(\mathbf{X}, \mathbf{R})$, further examples are referred to Tab.2 of (Xu, 2008a). A less constrained link is provided by a Hessian based conversation, i.e., a conservation on local convexity, which is applicable to \mathbf{Y}, Θ of real variables. This is a type of local information conservation, e.g., Fisher information conservation. The most relaxed conservation is based on a collective measure instead of details, e.g., Shannon entropy.

Again, we use $S_k(\Theta_k)$ to denote a family of system specifications, with \mathbf{k} featured by the scale or complexity for representing \mathbf{R} , which is contributed by the scale \mathbf{k}_Y for representing \mathbf{Y} and an effective number standing for free parameters in Θ_k . An analogy of this Ying Yang system to the Chinese ancient Ying-Yang philosophy motivates to determine \mathbf{k} and Θ_k under **the best harmony principle**, mathematically to maximize

$$H(p || q, \mathbf{k}, \Xi) = \int p(\mathbf{R} | \mathbf{X}) p(\mathbf{X} | \mathbf{X}_N, h) \ln[q(\mathbf{X} | \mathbf{R})q(\mathbf{R})] d\mathbf{X} d\mathbf{R}. \quad (8)$$

On one hand, the maximization forces $q(\mathbf{X} | \mathbf{R})q(\mathbf{R})$ to match $p(\mathbf{R} | \mathbf{X})p(\mathbf{X} | \mathbf{X}_N, h)$. In other words, $q(\mathbf{X} | \mathbf{R})q(\mathbf{R})$ attempts to describe the data $p(\mathbf{X} | \mathbf{X}_N, h)$ in help of $p(\mathbf{R} | \mathbf{X})$, which uses actually $q(\mathbf{X}) = \int q(\mathbf{X} | \mathbf{R})q(\mathbf{R}) d\mathbf{R}$ to fit $p(\mathbf{X} | \mathbf{X}_N, h)$ not in a maximum likelihood sense but with a promising model selection nature. Due to a finite size of \mathbf{X}_N and structural constraint of $p(\mathbf{R} | \mathbf{X})$, this matching aims at (but may not really reach) $q(\mathbf{X} | \mathbf{R})q(\mathbf{R}) = p(\mathbf{R} | \mathbf{X})p(\mathbf{X} | \mathbf{X}_N, h)$. Still we get a trend at this equality by which $H(p || q, \mathbf{k}, \Xi)$ becomes the negative entropy, and its further maximization is minimizing the system complexity, which consequently provides a model selection nature on \mathbf{k} .

At the first glance, one may feel the formulae eqn.(8) somewhat familiar. In fact, its degenerated case with \mathbf{R} vanished leads to $H(p \parallel q) = \int p(\mathbf{X}) \ln q(\mathbf{X}) d\mathbf{X}$. With $p(\mathbf{X}) = p(\mathbf{X} | \mathbf{X}_N, h)$ at $h = 0$ and $q(\mathbf{X}) = q(\mathbf{X} | \Theta)$, maximizing this $H(p \parallel q)$ becomes $\max_{\Theta} \ln q(\mathbf{X}_N | \Theta)$, i.e., maximum likelihood (ML) learning. The situation with $p(\mathbf{X})$ beyond $p(\mathbf{X} | \mathbf{X}_N, h)$ was also explored in the signal processing literature, via $\min_{q(\mathbf{X})} - H(p \parallel q)$ under the name of Minimum Cross-Entropy (Minxent). It was noticed that $\min_{p(\mathbf{X})} - H(p \parallel q)$ leads to a singular result that $p(\mathbf{X}) = \delta(\mathbf{X} - \mathbf{X}^*)$, $\mathbf{X}^* = \arg \max_{\mathbf{X}} \ln q(\mathbf{X} | \Theta)$, which was regarded as irregular and not useful. Instead, efforts were turned to minimizing the classic Kullback–Leibler divergence $KL(p \parallel q) = H(p \parallel p) - H(p \parallel q)$ with respect to p , which pushes p to match q and thus the above singular result is avoided. Moreover, if p is given, $\min_{q(\mathbf{X})} KL(p \parallel q)$ is still equivalent to $\min_{q(\mathbf{X})} - H(p \parallel q)$. Thereafter, Minimum Cross-Entropy is usually used to refer this $\min_{q(\mathbf{X})} KL(p \parallel q)$.

Interestingly, with an inner representation \mathbf{R} considered in the BYY system, the scenario becomes different from the above classic situation. With $p(\mathbf{X}) = p(\mathbf{X} | \mathbf{X}_N, h)$ fixed, $\max_{p(\mathbf{X})} H(p \parallel q)$ is made with respect to only $p(\mathbf{R} | \mathbf{X})$, which is no longer useless but responsible to the promising least complexity nature discussed after eqn.(8). In other words, maximizing $\max_{p(\mathbf{X})} H(p \parallel q)$ with respect to unknowns not only in Ying part $q(\mathbf{X}, \mathbf{R})$ but also in Yang part $p(\mathbf{X}, \mathbf{R})$ makes the Ying-Yang system become a best harmony. Alternatively, we may regard such a mathematical formulation as an information theoretic interpretation of the ancient Ying Yang philosophy.

In implementation, $H(p \parallel q, \mathbf{k}, \Xi) = \int p(\Theta | \mathbf{X}_N) H(p \parallel q, \Theta, \mathbf{k}, \Xi) d\Theta$ can be approximated via a Taylor expansion of $H(p \parallel q, \Theta, \mathbf{k}, \Xi)$ around Θ^* up to the second order as follows:

$$\begin{aligned}
 H(p \parallel q, \mathbf{k}, \Xi) &= \int p(\Theta | \mathbf{X}_N) H(p \parallel q, \Theta, \mathbf{k}, \Xi) d\Theta \approx H(p \parallel q, \Theta^*, \mathbf{k}, \Xi) + 0.5 d_{\mathbf{k}}(\Xi), \\
 \Theta^* &= \arg \max_{\Theta} H(p \parallel q, \Theta, \mathbf{k}, \Xi), \quad d_{\mathbf{k}}(\Xi) = Tr[\Gamma \Omega(\Theta^*)] + (\zeta - \Theta^*)^T \Omega(\Theta^*, \Xi) (\zeta - \Theta^*), \\
 H(p \parallel q, \Theta, \mathbf{k}, \Xi) &= H_0(p \parallel q, \Theta, \mathbf{k}, \Xi) + \ln q(\Theta | \Xi), \\
 H_0(p \parallel q, \Theta, \mathbf{k}, \Xi) &= \int p(\mathbf{Y} | \mathbf{X}, \Theta_{y|x}) p(\mathbf{X} | \mathbf{X}_N, h) \ln[q(\mathbf{X} | \mathbf{Y}, \Theta_{x|y}) q(\mathbf{Y} | \Theta_y)] d\mathbf{X} d\mathbf{Y} + \ln q(\Theta | \Xi), \\
 \Omega(\Theta, \Xi) &= \nabla_{\Theta}^2 H(p \parallel q, \Theta, \mathbf{k}, \Xi), \quad \zeta = \int \Theta p(\Theta | \mathbf{X}_N) d\Theta, \quad \Gamma = \int (\Theta - \zeta)(\Theta - \zeta)^T p(\Theta | \mathbf{X}_N) d\Theta. \tag{9}
 \end{aligned}$$

The maximization of $H(p \parallel q, \mathbf{k}, \Xi)$ consists of an inner maximization that seeks the best harmony among the front layer Ying-Yang supported by a priori $q(\Theta | \Xi)$ and an outer maximization that seeks the best harmony of the entire Ying Yang system with $d_{\mathbf{k}}(\Xi)$ taken in consideration for the interaction between the front and back layers.

BYY HARMONY LEARNING: CHARACTERISTICS AND IMPLEMENTATIONS

Systematically considering two pathways and two domains coordinately as a BYY system under a probability theoretic ground and designing three system components under the principles of *least redundancy*,

divide-conquer, and *uncertainty conversation*, respectively. BYY harmony learning is different not only from cognitive science motivated adaptive resonance theory (Grossberg, 1976; Grossberg & Carpenter, 2002) and the least mean square error reconstruction based auto-association (Bourlard & Kamp, 1988) and LMSER self-organization (Xu, 1991 & 93), but also from those probability theoretic approaches that either merely consider a bottom-up pathway as the inverse of a top-down pathway (e.g., Bayesian approaches), or approximate such inverses for tackling intractable computations (e.g., Helmholtz machine, variational Bayes, and extensions). Mathematically implementing the Ying-Yang harmony philosophy, in a sense that Ying and Yang not only matches each other but also seeks a best match in a most compact way, by determining all unknowns in a BYY system via maximizing the functional of $H(p \parallel q, \mathbf{k}, \Xi)$, with the model selection nature explained after eqn.(8).

Readers are further referred to (Xu, 2007a) for a systematical overview on relations of Bayesian Ying Yang learning to a number of typical approaches for either or both of parameter learning and model selection, covering the following aspects:

- **Best Data Matching perspective**, for modeling a latent or generative model and involving ML, Bayesian, AIC, BIC, MDL, MML, marginal likelihood, etc;
- **Best Encoding perspective**, for encoding inner representation by a bottom-up pathway (or called a transformation / recognition / representative model) and involving INFOMAX, MMI based ICA approaches and extensions;
- **Two Pathway Perspective**, involving information geometry based em-algorithm, Helmholtz Machine, variational approximation, and bits-back based MDL, etc.
- **Optimal Information Transfer Perspective**, involving MDL, MML, and bits-back based MDL, etc.

The model selection nature of maximizing $H(p \parallel q, \mathbf{k}, \Xi)$ can also be observed from its gradient follow with the Yang path in a Bayesian structure $p(\mathbf{R} | \mathbf{X}) = q(\mathbf{X} | \mathbf{R})q(\mathbf{R}) / q(\mathbf{X})$, $q(\mathbf{X}) = \int q(\mathbf{X} | \mathbf{R})q(\mathbf{R})d\mathbf{R}$. It follows that we have

$$dH(p \parallel q, \mathbf{k}, \Xi) = \int p(\mathbf{R} | \mathbf{X})[1 + \delta_L(\mathbf{X}, \mathbf{R})]dL(\mathbf{X}, \mathbf{R})p(\mathbf{X} | \mathbf{X}_N, h)d\mathbf{X}d\mathbf{R},$$

$$\delta_L(\mathbf{X}, \mathbf{R}) = L(\mathbf{X}, \mathbf{R}) - \int p(\mathbf{R} | \mathbf{X})L(\mathbf{X}, \mathbf{R})d\mathbf{R}, \text{ and } L(\mathbf{X}, \mathbf{R}) = \ln[q(\mathbf{X} | \mathbf{R})q(\mathbf{R})] \quad (10a)$$

Noticing that $L(\mathbf{X}, \mathbf{R})$ describes the fitness of an inner representation \mathbf{R} on the observation \mathbf{X} , we observe that $\delta_L(\mathbf{X}, \mathbf{R})$ indicates whether the considered \mathbf{R} fits \mathbf{X} better than the average of all the possible choices of \mathbf{R} . Letting $\delta_L(\mathbf{X}, \mathbf{R}) = 0$, the rest of $dH(p \parallel q, \mathbf{k}, \Xi)$ is actually the updating flow of the M step in the EM algorithm for the maximum likelihood learning (McLachlan & Geoffrey, 1997). Usually $\delta_L(\mathbf{X}, \mathbf{R}) \neq 0$, i.e., the gradient flow $dL(\mathbf{X}, \mathbf{R})$ under all possible choices of \mathbf{R} is integrated via weighting not just by $p(\mathbf{R} | \mathbf{X})$ but also by a modification of a relative fitness measure $1 + \delta_L(\mathbf{X}, \mathbf{R})$. If $\delta_L(\mathbf{X}, \mathbf{R}) > 0$, updating goes along the same direction of the EM learning with an increased strength. If $0 > \delta_L(\mathbf{X}, \mathbf{R}) > -1$, i.e., the fitness is worse than the average and the current \mathbf{R} is doubtful, updating still goes along the same direction of the EM learning but with a reduced strength. When $-1 > \delta_L(\mathbf{X}, \mathbf{R})$, updating reverses the direction of the EM learning and actually becomes de-learning. In other words, the BYY harmony learning shares the same nature of RPCL learning but with an improvement that

there is no need on a pre-specified de-learning strength $\gamma > 0$, as previously discussed in Figure 3(A) on a special case of NRBF & ME with $\mathbf{R} = \{j, \Theta\}$, where $p(\mathbf{R} | \mathbf{X})[1 + \delta_L(\mathbf{X}, \mathbf{R})]$ is simplified into $p_{j,t}(1 + \delta h_{t,j})$ and $\delta_L(\mathbf{X}, \mathbf{R})$ into $\delta h_{t,j}$.

More specifically, the model selection nature of Bayesian Ying Yang learning possess the following favorable promising features.

- The conventional model selection approaches aim at model complexity conveyed at either or both of the level of structure S_k and the level of parameter set Θ . This task is difficult to estimate, usually resulting in some rough bounds. Bayesian Ying Yang learning considers not only the levels of S_k and Θ but also the level of short memory representation \mathbf{Y} by the front layer of BYY system in Figure 4(a). That is, the scale \mathbf{k} of the BYY system is considered with the part \mathbf{k}_y for representing \mathbf{Y} , while the rest part, contributed by S_k and Θ , is estimated via $\ln q(\Theta | \Xi)$ and $d_k(\Xi)$ in eqn.(9), which is along a line similar to the above conventional approaches. Interestingly, the part \mathbf{k}_y is modeled via $q(\mathbf{Y} | \Theta_y)$ in $H_0(p || q, \Theta, \mathbf{k}, \Xi)$, which is estimated more accurately than the rest part. Promisingly, the model selection problems of many typical learning tasks can be reformulated into selecting merely the \mathbf{k}_y part in a BYY system (Xu, 2005). Therefore, the resulted BYY harmony criterion $J(\mathbf{k})$ shown by the left one of Figure 4(b) may considerably improve the performances by typical model selection approaches, which has been shown by in experiments (Shi, 2008).
- Even interestingly, this \mathbf{k}_y part associates with a set $\tilde{\Theta}_y \subseteq \Theta_y$ of parameters on which there exists an indicator $\rho(\tilde{\Theta}_y)$. Maximizing $H_0(p || q, \Theta, \mathbf{k}, \Xi)$ will exert a force that pushes $\rho(\tilde{\Theta}_y) \rightarrow 0$, which means that its associated contribution to \mathbf{k}_y can be discarded. E.g., each j of k values associates with one α_j , we can discard a structure if its correspondent $\rho(\alpha_j) = \alpha_j \rightarrow 0$. As a result, k effectively reduces to $k - 1$. As illustrated on the right of Figure 4(b), $\alpha_j \rightarrow 0$ means its contribution to $\tilde{J}(k)$ is 0, and a number of such parameters becoming 0 result in that $\tilde{J}(k)$ has effectively no change on a range $[\hat{k}, \tilde{k}]$. Also, each dimension $y^{(i)}$ of $q(y | \theta_j^y)$ associates with its variance λ_j and this dimension can be discarded if $\lambda_j \rightarrow 0$. As illustrated beyond \tilde{k} on the right of Figure 4(b), such a parameter becoming 0 contributes to $\tilde{J}(k)$ by $-\infty$. As long as \mathbf{k}_y is initialized at a big enough value, \hat{k} can be found as an upper bound estimate of \mathbf{k}^* . That is, an automatic model selection is incurred during parameter learning. Details are referred to Sec.2.3 of (Xu, 2008a&b).
- The separated consideration of \mathbf{k}_y from the rest of \mathbf{k} also provides a general framework that integrates the roles of regularization and model selection, such that not only the automatic model selection mechanism on \mathbf{k}_y can avoid the previously mentioned disturbance by a regularization with an inappropriate priori $q(\Theta | \Xi)$, but also imprecise approximations caused by handling the integrals may be alleviated via regularization. Specifically, model selection is made via $q(\mathbf{Y} | \Theta_y)$ in Ying machine, while regularization is imposed in Yang machine via designing its structure under a uncertainty conservation principle given at the bottom of Figure 4 and making *data smoothing regularization* via $p(\mathbf{X}) = p(\mathbf{X} | \mathbf{X}_N, h)$ with $h \neq 0$ that takes a role similar to regularization strength, while the difficulty of the conventional regularization approaches on controlling this strength has been avoided because an appropriate $h \neq 0$ is also determined during maximizing $H(p || q, \Theta, \mathbf{k}, \Xi)$.

At a first glance, a scenario with $q(\mathbf{Y} | \Theta_y)$ is seemly involved also in several typical learning approaches, especially those with an EM like two pathway implementation, such as the EM algorithm implemented ML learning (Redner & Walker, 1984), information geometry based em-algorithm (Amari, 1995), Helmholtz Machine (Hinton, Dayan, Frey, & Neal, 1995; Dayan, 2002), variational approximation (Jordan, Ghahramani, Jaakkola, & Saul, 1999), the bits-back based MDL (Hinton & Zemel, 1994), etc. Actually, these studies have neither put $q(\mathbf{Y} | \Theta_y)$ in a role for describing \mathbf{k}_y nor sought for the above nature of automatic model selection. Instead, the role of $q(\mathbf{Y} | \Theta_y)$ in these studies is estimating $q(\mathbf{X}) = \int q(\mathbf{X} | \mathbf{Y}, \Theta_{x|y}) q(\mathbf{Y} | \Theta_y) d\mathbf{Y}$ in a ML sense or approximately, which is similar to the one in Bayesian Kullback Ying Yang (BKYY) learning that not only accommodates a number of typical statistical learning approaches (including these studies) as special cases but also provides a bridge to understand the relation and difference from the best harmony learning by maximizing $H(p || q, \Theta, \mathbf{k}, \Xi)$ in eqn.(8). Proposed in (Xu,1995), BKYY learning performs a best Ying Yang matching by minimizing:

$$KL(p || q, \mathbf{k}, \Xi) = \int p(\mathbf{R} | \mathbf{X}) p(\mathbf{X} | \mathbf{X}_N, h) \ln \frac{p(\mathbf{R} | \mathbf{X}) p(\mathbf{X} | \mathbf{X}_N, h)}{q(\mathbf{X} | \mathbf{R}) q(\mathbf{R})} d\mathbf{X} d\mathbf{R} - H(p || q, \mathbf{k}, \Xi), \quad (10b)$$

that is, a best Ying Yang harmony includes not only a best Ying Yang matching as a part but also minimizing the entropy or the complexity of a Yang machine. In other words, it seeks a Yang machine that not only best matches the Ying machine but also keeps itself in a least complexity.

Considering a learning system in a Ying-Yang pair naturally motivates to implement the maximization of $H(p || q, \Theta, \mathbf{k}, \Xi)$ or the minimization of $KL(p || q, \Theta, \mathbf{k}, \Xi)$ by an alternative iteration of

- **Yang step:** fixing all the unknowns in the Ying machine, we update the rest of the unknowns in the Yang machine (after excluding those common unknowns shared by the Ying machine);
- **Ying step:** fixing those just updated unknowns in the Yang step, we update all the unknowns in the Ying machine.

Not only this iteration is guaranteed to converge, but also it includes the well known EM algorithm and provides a general perspective for developing other EM-like algorithms.

Recalling eqn.(9), the maximization of $H(p || q, \Theta, \mathbf{k}, \Xi)$ can be approximately decoupled into an inner maximization for the best harmony among the front layer supported by a priori $q(\Theta | \Xi)$ and an outer maximization for interaction between the front and back layers. Thus, we have the following two stage implementation:

Stage I enumerate each $\mathbf{k} \in \mathbf{K}$, initialize $\Xi^{(0)}$ and iterate :

$$\begin{aligned} (a) \quad \Theta^{(t)} &= \arg \max / \text{incr}_{\Theta} H(p || q, \Theta, \mathbf{k}, \Xi^{(t-1)}), \\ (b) \quad \Xi^{(t)} &= \arg \max / \text{incr}_{\Xi} [H(p || q, \Theta^{(t)}, \mathbf{k}, \Xi) + d_{\mathbf{k}}(\Xi)], \quad \Delta \Theta_{\tau}^{(t)} = \Theta^{(t)} - \Theta^{(t-\tau)}, \quad t \geq , \\ &\quad \partial_{\mathbf{k}} \Xi = -n_f(\Theta_k) + \Delta \Theta_{\tau}^{(t)T} \Omega(\Theta^{(t)}, \Xi) \Delta \Theta_{\tau}^{(t)}, \quad \text{after convergence we get } \Theta^*, \Xi^*; \\ \text{Stage II} \quad \mathbf{k}^* &= \arg \min_{\mathbf{k}} J(\mathbf{k}), \quad J(\mathbf{k}) = \tilde{J}(\mathbf{k}) + 0.5 n_f(\Theta_k). \end{aligned} \quad (11a)$$

during which a previous estimate $\Theta^{(t-\tau)}$ is used as ζ with $\Gamma = -\Omega^{-1}(\Theta^*, \Xi)$, and thus $d_k(\Xi)$ is simplified into $\Delta\Theta_\tau^{(t)T}\Omega(\Theta^{(t)}, \Xi)\Delta\Theta_\tau^{(t)}$, where an integer $n_f(\Theta_k)$ denotes the number of free parameters in Θ_k . Being different from a classic two stage implementation, not only model selection is made at Stage II, but also automatic model selection occurs during implementing Stage I that is actually implemented by one Ying Yang iteration as above, from which we can further get detailed algorithms, e.g., Algorithm 2 in the previous section and Algorithm 3 in the next section, as further discussed in sequel.

When samples $\mathbf{X}_N = \{x_t\}_{t=1}^N$ are independent and identically distributed (i.i.d.), from $\mathbf{Y}_N = \{y_t, j_t\}_{t=1}^N$ and $q(\Theta | \Xi) = q(h)\prod_{j=1}^k q(\Theta_j | \Xi)$, $H_0(p || q, \Theta, \mathbf{k}, \Xi)$ in eqn.(11a) becomes $\sum_{t=1}^N \sum_{j=1}^k p(j | x_t, \theta_j^{y|x}) \left\{ \int p(y | x_t, \theta_j^{y|x}) G(x | x_t, h_x^2 I) \ln[q(x | y, \theta_j^{x|y}) q(y | \theta_j^y) \alpha_j] dx dy \right\}$ with $\alpha_j = q(j)$. By a Taylor expansion of $\ln[q(x | y, \theta_j^{x|y}) q(y | \theta_j^y) \alpha_j]$ with respect to x, y around x_t and y_t respectively, from which we further get

$$\begin{aligned} H_0(p || q, \Theta, \mathbf{k}, \Xi) &\approx \sum_{t=1}^N \sum_{j=1}^k p(j | x_t, \theta_j^{y|x}) H_t(\Theta_j), \\ H_t(\Theta_j) &= \pi(x_t, \eta(x_t | \theta_j^{y|x}), \Theta_j) - 0.5 Tr[h^2 \Pi_j^x + \Gamma_j^{y|x} \Pi_j^y] + \frac{1}{N} \ln q(h), \\ \Pi_j^x &= -\nabla_x^2 \ln q(x | y, \theta_j^{x|y}), \quad \Pi_j^y = -\nabla_y^2 \pi_t(y, \Theta_j) \quad \pi_t(y, \Theta_j) = \ln[q(x_t | y, \theta_j^{x|y}) q(y | \theta_j^y) \alpha_j], \\ \eta(x_t | \theta_j^{y|x}) &= \int y p(y | x_t, \theta_j^{y|x}) dy, \quad \Gamma_j^{y|x} = \int p(y | x_t, \theta_j^{y|x}) [y - \eta(x_t | \theta_j^{y|x})] [y - \eta(x_t | \theta_j^{y|x})]^T dy. \end{aligned} \quad (11b)$$

In the special case $\mathbf{Y}_N = \{j_t\}_{t=1}^N$, i.e., $\{y, j\} \rightarrow j$ without y , $H_t(\Theta_j)$ becomes $\pi(x, \Theta_j) - 0.5 Tr[h^2 \nabla_x^2 \ln q(x | \theta_j)] + \frac{1}{N} \ln q(h)$, $\pi(x, \Theta_j) = \ln[q(x | \theta_j) \alpha_j]$. Further considering the substitution $x \rightarrow x, z$ and thus $q(x | \theta_j) \rightarrow q(x | \phi_j) q(z | x, \theta_j)$ with $G(x | x_t, h_x^2 I) \rightarrow G(x | x_t, h_x^2 I) G(z | z_t(x_t), h_z^2 I)$ and $q(h) \rightarrow q(h_x) q(h_z)$, $H_t(\Theta_j)$ in eqn.(11b) is simplified into the one same as in Algorithm 2(a).

We consider the design of $p(j | x_t, \theta_x)$ according to a principle of uncertainty conversation between Ying-Yang, under the likelihood based measure given in Figure 4(c). From $p(j | x, \theta_x) \propto q(x | \phi_j) q(z | x, \theta_j) \alpha_j$ and the constraint $\sum_j p(j | x, \theta_x) = 1$, we have $p(j | x, \theta_x) = q(x | \phi_j) q(z | x, \theta_j) \alpha_j / \sum_j q(x | \phi_j) q(z | x, \theta_j) \alpha_j = p_t(j | \Theta)$, which concurs with the one in Algorithm 2(a) and Figure 3(a). In this setting, we further have

- $dH_0(p || q, \Theta, \mathbf{k}, \Xi)$ as in Algorithm 2 (b), based on which we update $\Theta^{new} = \Theta^{old} + \Delta\Theta$, $\Delta\Theta \propto dH_0(p || q, \Theta, \mathbf{k}, \Xi) / d\Theta$ with $q(\Theta_j | \Xi)$ ignored by letting $\ln q(\Theta_j | \Xi) = 0$, which leads to Algorithm 2 and Figure 3(a).
- $\eta_{j,t} = p_{j,t}(1 + \delta h_{t,j})$ modifies the EM learning by $1 + \delta h_{t,j}$ that either enhances or weaken $p_{j,t}$ depending on whether or not its fitness is better than the average. Even this $\eta_{j,t} = p_{j,t}(1 + \delta h_{t,j})$ will become negative if its fitness falls far below the average, and thus leads to de-learning, which acts as an improvement of RPCL learning.
- Maximization of $H_0(p || q, \Theta, \mathbf{k}, \Xi)$ pushes $\sum_{t=1}^N \sum_{j=1}^k p(j | x_t, \theta_j^{y|x}) \ln \alpha_j \rightarrow N \sum_{j=1}^k \alpha_j \ln \alpha_j$ with $N \alpha_j = \sum_{t=1}^N p(j | x_t, \theta_j^{y|x})$. Further more, $N \sum_{j=1}^k \alpha_j \ln \alpha_j \leq 0$ approaches its upper bound

with some $\alpha_j \rightarrow 0$ if the j -th basis function or expert is redundant, and its contribution $N\alpha_j \ln \alpha_j$ becomes 0. In other word, we get an example that concurs with the previously discussed feature, i.e., $\tilde{J}(k)$ remains unchanged on $[\hat{k}, \tilde{k}]$, as illustrated on the right of Figure 4(b).

The above scenario can be directly extended to the case $\mathbf{Y}_N = \{y_t, j_t\}_{t=1}^N$ with

$$q(x | \phi_j) = \int q(x, y, \theta_j^{x|y}) q(y | \theta_j^y) dy \quad (11c)$$

put into $H_t(\Theta_j)$ in Algorithm 2(a).

In fact, the resulted model conceptually has no big difference from NRBF, ENRBF, ME, etc. The difference lays in that the basis function bases on the marginal density $q(x | \phi_j)$.

More generally, we can put the substitution $x \rightarrow x, z$ and its induced substitutions $q(x | y, \theta_j^{x|y}) \rightarrow q(x, z | y, \theta_j^{x,z|y}) = q(z | x, y, \theta_j^{z|x,y}) q(x | y, \theta_j^{x|y})$, $G(x | x_t, h^2 I) \rightarrow G(x | x_t, h_x^2 I) G(z | z_t(x_t), h_z^2 I)$ and $q(h) \rightarrow q(h_x) q(h_z)$ directly into eqn.(11b), resulting in

$$H_0(p || q, \Theta, \mathbf{k}, \Xi) \approx \sum_{t=1}^N \sum_{j=1}^k p(j | x_t, z_t, \theta_{y|x}) H_t(\Theta_j), \quad (11d)$$

$$H_t(\Theta_j) = \pi_t(\Theta_j) - \frac{1}{2} \text{Tr}[h_x^2 (\Pi_j^x + \Pi_j^{z|x}) + h_z^2 \Pi_j^z + \Gamma_j^{y|x} \Pi_j^y] + \frac{1}{N} \ln \{q(h_x) q(h_z)\},$$

$$\Pi_j^{z|x} = -\nabla_x^2 \ln q(z | \xi, \theta_j^{z|\xi}), \quad \Pi_j^z = -\nabla_z^2 \ln q(z | \xi, \theta_j^{z|\xi}), \quad \xi = \{x, y\},$$

$$\pi_t(\Theta_j) = \pi_t(\eta(x_t | \theta_j^{y|x}), \Theta_j), \quad \pi_t(y, \Theta_j) = \ln[q(z_t | \xi, \theta_j^{z|\xi}) q(x_t | y, \theta_j^{x|y}) q(y | \theta_j^y) \alpha_j].$$

Π_j^x , $\eta(x_t | \theta_j^{y|x})$ and $\Gamma_j^{y|x}$ are same as in eqn.(11b), from which and eqn.(11c) we can derive Algorithm 3 to be introduced in the next section. Moreover, in additional to the feature of pushing an extra $\alpha_j \rightarrow 0$, the maximization of $H_0(p || q, \Theta, \mathbf{k}, \Xi)$ in eqn.(11d) pushes $\pi_t(\Theta_j)$ to increase, which then pushes $\ln q(y | \theta_j^y)$ to increase. If there is a redundant dimension $y^{(i)}$, the corresponding $q(y^{(i)} | \theta_j^y)$ is pushed towards $\delta(y^{(i)} - c^{(i)})$ with its variance $\lambda_j \rightarrow 0$, which contributes to $\tilde{J}(k)$ by $-\infty$. In other word, we also get an specific example that concurs with the previously discussed feature that $\tilde{J}(k)$ tends $-\infty$, as illustrated beyond \tilde{k} on the right of Figure 4(b).

SUBSPACE BASED FUNCTIONS AND CASCADED EXTENSIONS

In many practices, there is only a finite size of training samples distributed in small dimensional subspaces, instead of scattering over all the dimensions of the observation space. These subspace structures can not be well captured by considering $q(x | \phi_j) = G(x | \mu_j, \Sigma_j)$ only. Moreover, there are too many free parameters in Σ_j , which usually leads to poor performances. Towards the problems, we consider $q(x | \phi_j)$ via subspace by independent factor analysis as shown in Figure 5, where observed samples are regarded as generated from a subspace with independent factors distributed along each coordinate of an inner m dimensional representation $y = [y^{(1)}, \dots, y^{(m)}]$. Typically, we consider $q(y^{(i)} | \theta_j^y)$ of a Gaussian

for a real $y^{(i)}$ and of a Bernoulli for a binary $y^{(i)}$.

By eqn. (10c) we get $q(x | \phi_j)$ to be put into eqn.(5) for extensions of basis functions or gating nets $g_j(x, \phi)$, which leads to subspace based extensions of ME and RBF networks. Correspondingly, $f(x)$ in eqn.(3) has been extended from a combination of functions supported on radial bases to a combination of functions supported on subspace bases, which are thus called subspace based functions (SBF). As discussed after eqn.(11c), learning of SBF can still be made by either Algorithm 1 or 2, with the updating formulae for $q(x | \phi_j)$ revised accordingly. Also, to facilitate computing we get $\alpha_j q(x | \phi_j) = \rho_t(\Theta_j)$ approximately by

$$\rho_t(\Theta_j) = \int e^{\pi_t(y, \Theta_j)} dy \approx (2\pi)^{0.5m_j} |\Pi_j^y|^{0.5} \exp[\pi(\eta(x_t | \theta_j^{y|x}), \Theta_j)], \quad \Pi_j^y \text{ as in eqn.(11b)}, \quad (12)$$

where ' \approx ' becomes '=' when $q(x, y | \theta_j^{x|y})$ and $q(y | \theta_j^y)$ are both Gaussian.

As before, an appropriate number k can be determined by automatic model selection during implementing Algorithm 2. In addition to k , we need an appropriate dimension m_ℓ for each subspace too, which can not be performed by Algorithm II. Though the problem may be handled in help of a two stage implementation, the effect of a finite number of samples will more seriously deteriorate the evaluation on the values of $J(\Theta_k)$ because we need to determine $k+1$ integers (i.e., k plus $m_\ell, \ell = 1, \dots, k$).

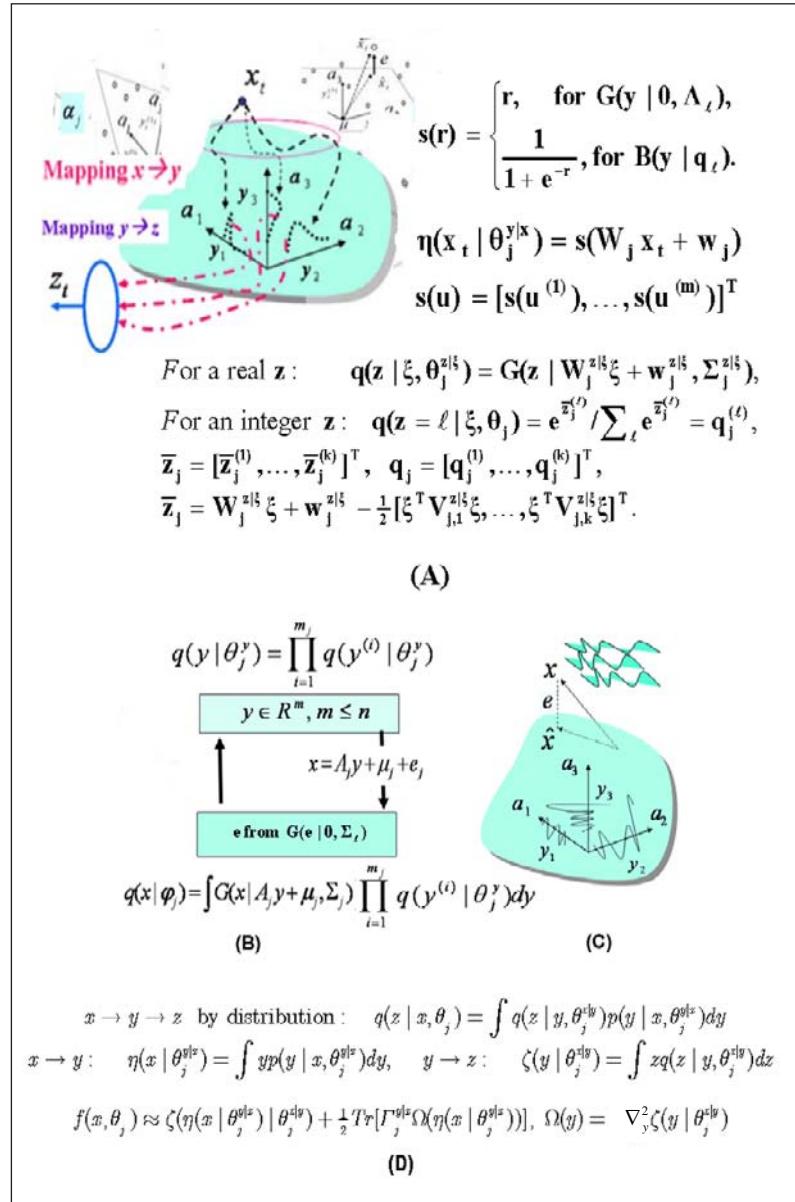
Alternatively, instead of implementing a direct mapping $x \rightarrow z$ by $q(z_t | x_t, \theta_j^{z|x})$ or $f_j(x, \theta_j^{z|x})$, extensions can also be made in help of the hidden factors y by $q(z_t | \xi, \theta_j^{z|\xi}) = q(z_t | y, \theta_j^{z|y})$, as shown in Figure 5. Two cascade mappings $x \rightarrow y$ and $y \rightarrow z$ replace the direct map $x \rightarrow z$ by $f_j(x, \theta_j^{z|x})$ or $q(z_t | x_t, \theta_j^{z|x})$. The mapping $x \rightarrow y$ acts as feature extraction, and thus the cascade $x \rightarrow y \rightarrow z$ can discard redundant parts, which is called cascaded subspace based functions. In a regression task, the cascaded $x \rightarrow y \rightarrow z$ is exactly a compound function of the regressions for $x \rightarrow y$ and $y \rightarrow z$ when the latter is linear. In other cases, an approximation is obtained from a Taylor expansion of $\zeta(y | \theta_j^{z|y})$ around $\eta(x | \theta_j^{y|x})$ up to the second order.

However, it is no longer applicable to use eqn.(11c) to get $q(x | \phi_j)$ in eqn.(5) for $g_j(x, \phi)$. Instead, we consider $q(x | \phi_j)q(z | x, \theta_j) = \int q(x | y, \theta_j^{x|y})q(y | \theta_j^y)q(z | \xi, \theta_j^{z|\xi})dy$ and get $p(j | x, z_t)$ from eqn.(6b). To facilitate computing, we can also get $\alpha_j q(x | \phi_j)q(z | x, \theta_j) = \rho_t(\Theta_j)$ approximately by eqn.(12) with $\pi_t(y, \Theta_j)$ extended accordingly. That is, we have

$$p(j | x, z) = \rho_t(\Theta_j) / \sum_{j=1}^k \rho_t(\Theta_j), \quad \text{with } \rho_t(\Theta_j) \text{ by eqn.(12) and } \pi_t(y, \Theta_j) \text{ by eqn.(11d)}. \quad (13)$$

Learning is made to maximize $H_0(p || q, \Theta, \mathbf{k}, \Xi)$ in eqn.(11d), from which we obtain an adaptive algorithm as summarized in Algorithm 3, for implementing learning with an appropriate number k and an appropriate dimension m_ℓ of each subspace determined automatically. According to three typical cases of $q(z | \xi, \theta_j^{z|\xi})$, Algorithm 3 performs one of the tasks shown in Figure 6.

Figure 5. Subspace based independent factor analyses and extensions to cascaded subspace based functions



- Let $q(z | \xi, \theta_j^{z|\xi}) = 1$ (i.e., ignored), it degenerates into the adaptive algorithms for learning on either a local factor analysis (LFA) given in Table 3 in (Xu, 2008) or a local binary factor analysis (BFA).
- When $q(z | \xi, \theta_j^{z|\xi}) = G(z | W_j^{z|\xi} \xi + w_j^{z|\xi}, \Sigma_j^{z|\xi})$, it performs a regression that consists of a number of local functions. For the SBF case with $q(z | x, \theta_j^{z|x}) = G(z | W_j^{z|x} x + w_j^{z|x}, \Sigma_j^{z|x})$, each local linear function is $z = W_j^{z|x}(x - \mu_j) + w_j^{z|x}$, and the mapping $x \rightarrow z$ is $\sum_{j=1}^k p(j | x_t, \Theta)[W_j^{z|x}(x_t - \mu_j) + w_j^{z|x}]$. In this case, Algorithm 3 shares with Algorithm 2 the

Figure 6. Typical terms in Algorithm 3

	Gaussian	Discrete $z = 1, \dots, k$
	$q(z \xi, \theta_j^{z \xi}) = G(z W_j^{z \xi} \xi + w_j^{z \xi}, \Sigma_j^{z \xi})$ $\bar{z}_j = W_j^{z \xi} \xi + w_j^{z \xi} - \frac{1}{2} [\xi^T V_{j,1}^{z \xi} \xi, \dots, \xi^T V_{j,k}^{z \xi} \xi]^T$	$q(z = \ell \xi, \theta_j) = e^{\bar{z}_j^{(\ell)}} / \sum_\ell e^{\bar{z}_j^{(\ell)}} = q_j^{(\ell)}$, $W_j^{z \xi T} (e_\ell - q_{j,t}) - (V_{j,\ell}^{z \xi} - \sum_k q_{j,t}^{(k)} V_{j,k}^{z \xi}) \xi$
$g_j^{z \xi} = \nabla_\xi \ln q(z \xi, \theta_j^{z \xi})$	$W_j^{z \xi T} \Sigma_j^{z \xi-1} e_t^{z \xi}$, $e_t^{z \xi} = z_t - W_j^{z \xi} \xi$	$W_j^{z \xi T} (e_\ell - q_{j,t}) - (V_{j,\ell}^{z \xi} - \sum_k q_{j,t}^{(k)} V_{j,k}^{z \xi}) \xi$
$\Pi_j^{z \xi} = -\nabla_\xi^2 \ln q(z \xi, \theta_j^{z \xi})$	$W_j^{z \xi T} \Sigma_j^{z \xi-1} W_j^{z \xi}$	approximately $W_j^{z \xi T} (\text{diag}[q_{j,t}] - q_{j,t} q_{j,t}^T) W_j^{z \xi}$ + $V_{j,\ell}^{z \xi} - \sum_k q_{j,t}^{(k)} V_{j,k}^{z \xi}$
$\Pi_j^z = -\nabla_z^2 \ln q(z \xi, \theta_j^{z \xi})$	$\Sigma_j^{z \xi-1}$	0
Data smoothing	h_z^2 estimated	simply setting $h_z^2 = 0$

(A)

	Gaussian	Discrete
$q(z y, \theta_j^{z y})$	$G(z W_j^{z y} y + w_j^{z y}, \Sigma_j^{z y})$	$q(z y, \theta_j) = q(z = \ell y, \theta_j)$
$e_t^{z y}$	$z_t - (W_j^{z y \text{ old}} y_{j,t}^* + w_j^{z y \text{ old}})$	$e_\ell - q_{j,t}$
Ψ_q	I	$\text{diag}[q_{j,t}] - q_{j,t} q_{j,t}^T$
$\Sigma_j^{z y}$ $V_{j,i}^{z y}$ $i = 1, \dots, k$	$\Sigma_j^{z y} = S_j^{z y} S_j^{z y T}$, $S_j^{z y \text{ new}} = S_j^{z y \text{ old}} + \Delta S_j^{z y}$, $\Delta S_j^{z y} \propto \eta_{j,t} G_j^{z y} S_j^{z y \text{ old} - T}$, $G_j^{z y} = e_{j,t}^{z y} e_{j,t}^{z y T} - \Sigma_j^{z y \text{ old}}$ $+ W_j^{z y \text{ old}} \Gamma_{j,t}^{y x} W_j^{z y \text{ old} T} + 0.5 h_z^2 I$	$V_{j,i}^{z y} = Q_{j,i}^{z y} Q_{j,i}^{z y T}$, $Q_{j,i}^{z y \text{ new}} = Q_{j,i}^{z y \text{ old}} + \Delta Q_{j,i}^{z y}$, $\Delta Q_{j,i}^{z y} \propto -\eta_{j,t} (\delta_{i\ell} - q_{j,t}^{(k)})$ $[\Gamma_{j,t}^y + (\delta_{i\ell} - q_{j,t}^{(k)}) \Gamma_{j,t}^{y x}] Q_{j,i}^{z y \text{ old} - T}$
	$w_j^{z y \text{ new}} = w_j^{z y \text{ old}} + \Delta w_j^{z y}$, $\Delta w_j^{z y} \propto \eta_{j,t} e_t^{z y}$,	
	$W_j^{z y \text{ new}} = W_j^{z y \text{ old}} + \Delta W_j^{z y}$, $\Delta W_j^{z y} \propto \eta_{j,t} (e_t^{z y} y_{j,t}^{* T} - \Psi_q^{old} W_j^{z y} \Gamma_{j,t}^{y x})$	

(B)

same equations for updating $q(z_t | x_t, \theta_j^{z|x})$ by Figure 3(C)&(D). For a cascaded SBF case with $q(z | y, \theta_j^{z|y}) = G(z | W_j^{z|y} y + w_j^{z|y}, \Gamma_j^{z|y})$, a local mapping is made by $\eta(x | \theta_j^{y|x}) = s(W_j x + w_j)$ that performs a local dimension reduction and a regression $z = W_j^{z|y} y + w_j^{z|y}$, with $x \rightarrow z$ by $\sum_{j=1}^k p(j | x_t, \Theta) [W_j^{z|y} s(W_j x + w_j) + w_j^{z|y}]$. Being different from Algorithm 2 that makes automatic selection only on k , learning by Algorithm 3 on the part for $y \rightarrow z$ is coupled with the part for $x \rightarrow y$, during which automatic model selection is obtained on the dimension of each subspace, as previously explained after eqn.(11d).

- When $q(z | y, \theta_j) = q(z = \ell | y, \theta_j)$, $\ell = 1, \dots, k$, it performs a classification $x \rightarrow z$ into one of labels. In help of those terms in Figure 6, the task is made by a group of classifiers with a local

feature extraction by $\eta(x \mid \theta_j^{y|x}) = s(W_j x + w_j)$.

Simplified variants of Algorithm 3 may be considered with degenerated settings on one or more of $h_x = 0$, $h_z = 0$, $\Gamma_j^{y|x} = 0$ that actually remove certain regularization on learning. Specifically, $h_x = 0$ shuts down smoothing regularization on samples of x , and $h_z = 0$ shuts down the one on samples of z , while $\Gamma_j^{y|x} = 0$ shuts down a regularization on the domain of y . Also, Algorithm 3 may degenerate into EM or RPCL with a degenerated setting $h_x = 0$ by choosing its corresponding allocating scheme in Figure 3(A).

The cases that the variable $y^{(i)}$ on each coordinate of a subspace is binary by a Bernoulli are usually encountered in practical problems that encodes each input x into binary codes for subsequent processing (Heinen, 1996; Treier & Jackman, 2002). The cascaded mapping $x \rightarrow y \rightarrow z$ implements $f_j(x, \theta_j^{z|x}) = W_j^{z|y} s(Wx + w_j) + w_j^{z|y}$ that is a three layer forward net. In other words, the special case $k = 1$ leads us to a classic three layer forward network. Being different from both the least square back propagation learning (Rumelhart, Hinton, & Williams, 1986) and the EM algorithm based ML learning (Ma, Ji, & Farmer, 1997), there is favorably an automatic model selection on the number m_ℓ of hidden units during implementing Algorithm 3 (shown in Figure 7). Moreover, a general case $k > 1$ leads to $f(x, \theta^{z|x}) = \sum_{j=1}^k p(j \mid x_t, \Theta) f_j(x, \theta_j^{z|x})$, i.e., a mixture of experts with each expert in a three layer forward networks. Sharing a hidden representation y in Figure 5(A) by experts and its gate, the number of free parameters is reduced. This mixture of experts is different from the conventional way of using one three layer forward networks as each expert in either (Jacobs, Jordan, Nowlan, & Hinton, 1991) or (Xu, Jordan & Hinton, 1995), where $q(z \mid \xi, \theta_j^{z|\xi}) = q(z \mid x, \theta_j^{z|x})$ is implemented via a three layer forward networks as a whole.

In the cases with a binary vector y , the integral $\int dy$ becomes a summation \sum_y over all the 2^{m_ℓ} terms, which becomes impractical computationally. Conceptually, the technique of using Taylor expansion in eqn.(11b) and eqn.(11d) is not applicable to y of a binary vector. Still, we can get eqn. (11b) and eqn.(11d) by rewriting $\pi_t(y, \Theta_j)$ in a format $b_0 + b_1[y - Ey] + b_2[y - Ey][y - Ey]^T$, and we replace $\int dy$ by \sum_y to get $\Gamma_j^{y|x}$ in eqn.(11b) and $\rho_t(\Theta_j)$ in eqn.(13). Moreover, the task of getting $y_{j,t}^* = \arg \max_y \pi_t(y, \Theta_j)$ is also conceptually a NP hard 0-1 programming. In Algorithm 3, it is approximately handled as suggested in (Table II, Xu, 2001b) under the name of fixed posteriori approximation, i.e., solving $\nabla_y \pi_t(y, \Theta_j) = 0$ as if each $y^{(i)}$ is real and then is rounded into binary. With an extra computing for solving a nonlinear equation of a scalar variable, a further improvement can be obtained by adopting the canonical dual approach (Fang, Gao, Shu, & Wu, 2008; Sun, Tu, Gao, & Xu, 2009).

In addition to getting $\Gamma_j^{y|x}$ in eqn.(11b) by $\Gamma_j^{y|x} = \varepsilon_{j,t} \varepsilon_{j,t}^T$ in Algorithm 3, we also have a degenerated choice and a generalized choice as follows:

- a degenerated choice is considering $\Gamma_j^{y|x} = \text{diag}[\gamma_{j,t}^{(1)}, \dots, \gamma_{j,t}^{(m_j)}]$, $\gamma_{j,t}^{(i)} = s(\bar{y}_t^{(i)})(1 - s(\bar{y}_t^{(i)}))$, which is equivalently considering a conditional independent distribution $q(y \mid x, \theta_j^{y|x}) = \prod_{i=1}^{m_j} s(\bar{y}_t^{(i)})^{y^{(i)}} (1 - s(\bar{y}_t^{(i)}))^{1-y^{(i)}}$ $\bar{y}_t = W_j^{y|x} x_t + w_j$
- a generalized choice is estimating:

Figure 7. Algorithm 3: Adaptive algorithms for LFA, cascade SBF and their binary variants

With $q(x_t y, \theta_{xy,j}) = G(x_t A_j y + \mu_j, \Sigma_j)$ and $q(z_t \xi, \theta_j^{z x})$ in one of choices in Fig.6(A), eq.(10d) becomes $H_0(p \ q, \Theta, k, \Xi) \approx \sum_{t=1}^N \sum_{j=1}^k p(j x_t, \theta_j^{y x}) H_t(\Theta_j)$.
$H_t(\Theta_j) = \pi(x_t, \eta(x_t \theta_j^{y x}), \Theta_j) - \frac{1}{2} \text{Tr}[\Gamma_j^{y x} \Pi_j^y + h_x^2 (\Sigma_j^{-1} + \Pi_j^{z x}) + h_z^2 \Pi_j^z]$ $+ \frac{1}{N} \ln[q(h_x)q(h_z)], \quad \pi_t(y, \Theta_j) = \ln[\alpha_j G(x_t A_j y + \mu_j, \Sigma_j)q(y \theta_j^y)q(z_t \xi, \theta_j^{z x})]_{\xi=y \text{ or } x}$ $\Pi_j^y = \nabla_y^2 \pi_t(y, \Theta_j) = A_j^T \Sigma_j^{-1} A_j + \Pi_j^{zy} + \Delta_{\Pi}^{yy}, \quad \text{where we have}$
<ul style="list-style-type: none"> • $q(y \theta_j^y) = \begin{cases} G(y \mu_j^y, \Lambda_j), & \Lambda_j \text{ is diagonal, Gaussian,} \\ B(y q_j) = \prod_{i=1}^m (1 - q_i^{(0)})^{1-y^{(0)}} q_i^{y^{(0)}}, & \text{Bernoulli.} \end{cases} \quad \Delta_{\Pi}^{yy} = \begin{cases} \Lambda_j^{-1}, & \text{for } G(y \mu_j^y, \Lambda_j), \\ 0, & \text{for } B(y q_j). \end{cases}$ • $\Gamma_j^{y x} = \varepsilon_{j,t} \varepsilon_{j,t}^T, \quad \varepsilon_{j,t} = \eta(x_t \theta_j^{y x}) - y_{j,t}^*, \quad y_{j,t}^* = \text{argmax}_{\ell} \pi_\ell(y, \Theta_j), \quad \text{from the rationale}$ that the possible upmost bound of $H_0(p \ q, \Theta, k, \Xi)$ is got at $p(y x_t, z_t, j) = \delta(y - y_{j,t}^*)$. • $p(j x_t, \theta_j^{y x}), \eta(x_t \theta_j^{y x}), \Pi_j^{z x}, \Pi_j^z$ are given in Fig.6(A).
Maximization of $H_0(p \ q, \Theta, k, \Xi)$ is implemented via its gradient flow $\nabla_{\Theta} H_0(p \ q, \Theta, k, \Xi) =$ $\nabla_{[h_x, h_z]} \ln[q(h_x)q(h_z)] + \sum_{t=1}^N \sum_{j=1}^k p_{j,t} (1 + \delta h_{j,t}) \nabla_{\Theta_j} \pi(x_t, \eta(x_t \theta_j^{y x}), \Theta_j) -$ $\frac{1}{2} \sum_{t=1}^N \sum_{j=1}^k p_{j,t} \{ \nabla_{\Theta_j} \text{Tr}[\Gamma_j^{y x} \Pi_j^y + h_x^2 (\Sigma_j^{-1} + \Pi_j^{z x}) + h_z^2 \Pi_j^z] + \delta h_{j,t} \nabla_{\Theta_j} \ln \Pi_j^y \}$ <p>by iterating the following two steps until its convergence:</p>
YANG STEP: get $p_{j,t} = p(j z_t, x_t)$ and $\delta h_{j,t} = H_t(\Theta_j) - \sum_{\ell=1}^k p_{\ell,t} H_t(\Theta_\ell)$, <ul style="list-style-type: none"> • Solve $\nabla_y \pi_t(y, \Theta_j) = 0$, i.e. $A_j^T \Sigma_j^{-1} (x_t - A_j y + \mu_j) + \Lambda_j^{-1} (y - \mu_j^y) + g_j^{zy} = 0$ with g_j^{zy} in Tab.2(A), which is linear of y (or approximately for a discrete z). • update $W_j^{\text{new}} = W_j^{\text{old}} + \Delta W_j, \quad w_j^{\text{new}} = w_j^{\text{old}} + \Delta w_j, \quad \Delta W_j \propto p_{j,t} \varepsilon_{j,t} x_t^T, \quad \Delta w_j \propto \varepsilon_{j,t}$. <p>YING STEP get $\alpha_j^{\text{new}} = e^{\varepsilon_j^{\text{old}} + \Delta c_j} / \sum_{\ell} e^{\varepsilon_{\ell}^{\text{old}} + \Delta c_{\ell}}$, same as Step (a) in Algorithm II,</p> <p>If a $\alpha_j \rightarrow 0$, discard the corresponding structure and its Θ_j.</p> <p>(a) $\mu_j^{\text{new}} = \mu_j^{\text{old}} + \Delta \mu_j, \quad \Delta \mu_j \propto p_{j,t} (1 + \delta h_{j,t}) e_t^{zy}, \quad e_t^{zy} = x_t - (A_j^{\text{old}} y_{j,t} + \mu_j^{\text{old}}),$ $A_j^{\text{new}} = A_j^{\text{old}} + p_{j,t} \Delta A_j (I - A_j^{\text{old}} A_j^{\text{old}T}), \quad \Delta A_j \propto (1 + \delta h_{j,t}) e_t^{zy} y_{j,t}^T - A_j^{\text{old}} (\Gamma_{j,t}^{y x} + \delta h_{j,t} \Pi_j^{y-1}),$ $\Sigma_j^{\text{new}} = S_j S_j^T, \quad S_j^{\text{new}} = S_j^{\text{old}} (I + p_{j,t} S_j^{\text{old}} \Sigma_j^{\text{old}} \Delta \Sigma_j^{\text{old}} S_j^{\text{old}T}),$ $\Delta S_j \propto (1 + \delta h_{j,t}) (e_{j,t}^{zy} e_{j,t}^{zyT} - \Sigma_j^{\text{old}}) + A_j^{\text{old}} (\Gamma_{j,t}^{y x} + \delta h_{j,t} \Pi_j^{y-1}) A_j^{\text{old}T} + h_x^2 I.$</p> <p>(b) Get $y_{j,t} = \eta(x_t \theta_j^{y x})$, update $G(y \mu_j^y, \Lambda_j)$ by $\Lambda_j = D_j D_j^T, \quad D_j^{\text{new}} = (I + \Delta D_j) D_j^{\text{old}},$ $\Delta D_j \propto p_{j,t} \text{diag}[(1 + \delta h_{j,t}) (e_{j,t}^y e_{j,t}^{yT} - \Lambda_j^{\text{old}}) + \Gamma_{j,t}^{y x}], \quad e_{j,t}^y = y_{j,t} - \mu_j^y \text{ with } \mu_j^y = 0;$ Update $B(y q_j)$ by $q_j^{\text{new}} = 1 / (1 + e^{-p_j^{\text{new}}}), \quad \beta_j^{\text{new}} - \beta_j^{\text{old}} \propto p_{j,t} (1 + \delta h_{j,t}) (y_{j,t} - q_j)$. If $\lambda_j^{\text{old}} \rightarrow 0$ or $q_j^{\text{old}} (1 - q_j^{\text{old}}) \rightarrow 0$, discard the corresponding dimension y_j^0.</p> <p>(c) $\eta_{j,t} = 1 + \delta h_{j,t}$, update $q(z y, \theta_j^{zy})$ by Fig.6(B) or $q(z x, \theta_j^{zx})$ by Fig.3(C) & (D).</p> <p>(d) $h_z^{\text{new}} = h_z^{\text{old}} + \Delta h_z, \quad h_x^{\text{new}} = h_x^{\text{old}} + \Delta h_x$ same as Step (d) in Algorithm II.</p>

$$\Gamma_j^{y|x} = \frac{1}{\# N(y_{j,t}^*)} \sum_{y \in N(y_{j,t}^*)} [y - \eta(x_t | \theta_j^{y|x})] / [y - \eta(x_t | \theta_j^{y|x})]^T, \quad y_{j,t}^* = \arg \max_y \pi_t(y, \Theta_j), \quad (14)$$

where $N(y_{j,t}^*)$ consists of $y_{j,t}^*$ and those values with a κ bit distance away, e.g., $\kappa = 1$.

FUTURE TRENDS

A further direction is extending SBF models in order to model temporal relations among data, via providing $q(y^{(i)}|\omega^{(i)})$ with a temporal structure such that temporal structure underlying samples of x can be projected to the temporal structures of y , as shown Figure 5(C). There have been three types of efforts. The straightforward one is to let each $f_j(x, \theta_j)$ being a regression of past samples $f_j(\{x_{t-\tau}\}_{\tau=1}^\kappa, \theta_j^{z|x})$, which has no difference from $f_j(x, \theta_j)$ by regarding $\{x_{t-\tau}\}_{\tau=1}^\kappa$ as a vector. The second is embedding temporal structure

$$\alpha_t = Q\alpha_{t-1}, \quad \alpha_t = [\alpha_{t,1}, \dots, \alpha_{t,k}]^T, \quad Q = [q_{ji}], \quad 0 \leq q_{ji} \leq 1, \quad \sum_{i=1}^k q_{ji} = 1, \quad (15)$$

into each priori $0 \leq \alpha_j$, $\sum_j \alpha_j = 1$. From $t-1$ to t , a new α_t is computed from its past α_{t-1} , and modulates the gate $g_j(x, \phi)$ in eqn.(5) to vary as time goes. For learning Q , we modify Step (a) of Algorithm 2&3, in either of the two ways shown in Figure 8.

Recursive Updating

$$\begin{aligned} \text{Update } Q \text{ by } q_{ji} = e^{\frac{c_{ji}}{\ell}} / \sum_\ell e^{\frac{c_{j\ell}}{\ell}} \text{ via } C = [c_{ji}] \text{ by } C^{new} = C^{old} + \Delta C, \quad \Delta C \propto \alpha_{t-1} g_\alpha^T - Q^T \text{diag}[g_\alpha], \\ \text{which comes from } Tr[g_\alpha^T d\alpha_t] \text{ via } d\alpha_t = dQ\alpha_{t-1} \text{ and } dQ^T = dC - \mathbf{1}[\mathbf{q}_1^T d\mathbf{c}_1, \dots, \mathbf{q}_k^T d\mathbf{c}_k], \\ Q^T = [\mathbf{q}_1, \dots, \mathbf{q}_k], \quad \mathbf{q}_j = [q_{j1}, \dots, q_{jk}]^T, \quad \text{and from} \\ Tr[g_\alpha^T d\alpha_t] = Tr[g_\alpha^T dQ\alpha_{t-1}] = Tr[g_\alpha \alpha_{t-1}^T dQ^T] = Tr[g_\alpha \alpha_{t-1}^T dC - \text{diag}[g_\alpha] Q dC]. \end{aligned} \quad (16a)$$

Asymptotic Updating

As $t \rightarrow \infty$, $\alpha_t \rightarrow \alpha$ we have $\alpha = Q\alpha$, thus $(I-Q)d\alpha = (dQ)\alpha$,

$$d\alpha = (I-Q)^{-1}(dQ)\alpha, \quad \text{from } Tr[g_\alpha^T (I-Q)^{-1} dQ\alpha] = Tr[g_\alpha^Q \alpha^T dC - \text{diag}[g_\alpha^Q] Q dC],$$

or we update $C^{new} = C^{old} + \Delta C$, $\Delta C \propto \alpha^{old} g_\alpha^{QT} - Q^{old} {}^T \text{diag}[g_\alpha^Q]$, $g_\alpha^Q = (I - Q^{old})^{-T} g_\alpha$. (16b)

Putting eqn.(15) into eqn.(5), we get an extension of alternative ME to a temporal one gated by a Hidden Markov chain. Initially, let $\alpha_0 = [\alpha_{0,1}, \dots, \alpha_{0,k}]^T$ we can get α_t from either $\alpha_t = Q^t \alpha_0$ or via $Q^t \alpha_{t-1}$ step by step, which makes the gate varies as time.

Figure 8. Modified updating equations for temporal subspaces

$q(x_t y_t, \phi_j) = G(x_t A_j y_t + \mu_j, \Sigma_j), \quad A_j^T A_j = I$	
$q(y \theta_j^\gamma) = \prod_{i=1}^{m_j} q_j^{(i)\theta_j^\gamma} (1 - q_j^{(i)})^{1-\theta_j^\gamma}$ At step (b) of Algorithm III, $q_j^{(i)}$ is replaced via modifying its learning accordingly.	$\begin{bmatrix} 1 - q_j^{(i)} \\ q_j^{(i)} \end{bmatrix}_t = \pi \begin{bmatrix} 1 - q_j^{(i)} \\ q_j^{(i)} \end{bmatrix}_{t-1}, \quad 0 \leq \pi_{j,0}^{(i)}, \pi_{j,1}^{(i)} \leq 1, \quad \pi = \begin{bmatrix} \pi_{j,0}^{(i)}, 1 - \pi_{j,0}^{(i)} \\ 1 - \pi_{j,1}^{(i)}, \pi_{j,1}^{(i)} \end{bmatrix}$ updating π in a same way as $\alpha_t = Q\alpha_{t-1}$ in eqn.(15)
$q(y \theta_j^\gamma) = G(y \mu_j^\gamma, \Lambda_j)$ at step (b) of Algorithm III is replaced by $G(y_t B_j y_{t-1}, \Lambda_j^\gamma)$ $y_t = B_j y_{t-1} + e_t^\gamma, \quad \Lambda_j^\gamma = E e_t^\gamma e_t^{\gamma T} \neq I$ but diagonal still, $E e_t^\gamma = 0, \quad E e_t^\gamma y_{t-1} = 0, \quad B_j = \Lambda_j^{\gamma 0.5} W_j \Lambda_j^{\gamma -0.5}, \quad W_j = \phi_j D_j \phi_j^T, \quad \phi_j^T \phi_j = I.$ $D_j = diag \frac{e^{d_j^{(i)}} - e^{-d_j^{(i)}}}{e^{d_j^{(i)}} + e^{-d_j^{(i)}}}_{i=1}^{m_j}$ for a real $-\infty < d_j^{(i)} < +\infty$	Regression parameterization $\mu_{j,t}^\gamma = B_j \mu_{j,t-1}^\gamma, \quad \mu_{j,t}^\gamma = E y_t$ For learning, we modify Step (b) of Algorithm III with $\Lambda_j = \Lambda_j^\gamma, \quad \mu_j^\gamma = B_j \mu_{j,t-1}^\gamma, \quad B_j^{new} = B_j^{old} + \Delta B_j, \quad \Delta B_j \propto p_{j,t} e_t^\gamma$. Marginalization equivalently $\Lambda_{j,t} = B_j \Lambda_{j,t-1} B_j^T + \Lambda_j^\gamma, \quad \Lambda_{j,t} = E y_t^T$ Initially $\Lambda_{j,0} = \Lambda_j^\gamma$, we have $\Lambda_{j,t} = \Lambda_j^\gamma + \sum_{\tau=1}^t B_j^\tau \Lambda_j^\gamma (B_j^\tau)^T = \Lambda_j^\gamma + \sum_{\tau=1}^t \Lambda_j^{\gamma 0.5} W_j^\tau \Lambda_j^{\gamma -0.5} \Lambda_j^\gamma \Lambda_j^{\gamma -0.5} W_j^\tau \Lambda_j^{\gamma 0.5} = \Lambda_j^\gamma + \Lambda_j^{\gamma 0.5} [\sum_{\tau=1}^t W_j^{2\tau}] \Lambda_j^{\gamma 0.5} = \Lambda_j^\gamma + \Lambda_j^{\gamma 0.5} \phi_j [\sum_{\tau=1}^t D_j^{2\tau}] \phi_j^T \Lambda_j^{\gamma 0.5}$, from which $\Lambda_{j,t} = \Lambda_j^{\gamma 0.5} \Sigma_j^D \Lambda_j^{\gamma 0.5} + \Lambda_j^\gamma$, $\Sigma_j^D = \phi_j D_j^{\Sigma} \phi_j^T, \quad D_j^{\Sigma} = [I - (i_\infty - 1) D_j^{2(t+1)}] (I - D_j^2)^{-1}$, where $i_\infty = 1$ if interested on an asymptotic behavior, otherwise, $i_\infty = 0$. Modifying Step (b) of Algorithm III with $\mu_j^\gamma = 0$, by G_{A_j} we update $\phi_j^{new} = \phi_j^{old} + \Delta \phi_j, \quad \Delta \phi_j \propto p_{j,t} G_{\phi_j} (I - \phi_j^{old} \phi_j^{old T}), \quad G_{\phi_j} = \Delta D_j \phi_j D_j^{\Sigma},$ $\Lambda_j^{new} = \Lambda_j^{old} + \Delta \Lambda_j^\gamma, \quad \Delta \Lambda_j^\gamma \propto p_{j,t} G_{\Lambda_j^\gamma}, \quad G_{\Lambda_j^\gamma} = \Delta D_j (\Lambda_j^\gamma + \Lambda_{j,t}),$ $D_j^{new} - D_j^{old} \propto p_{j,t} G_{D_j}, \quad G_{D_j}^T = [D_j^{\Sigma} D_j - i_\infty (t+1) D_j^{2t+1}] \phi_j^T \Delta D_j \phi_j$, which comes from putting $d\Lambda_{j,t} = d(\Lambda_j^{\gamma 0.5} \Sigma_j^D \Lambda_j^{\gamma 0.5} + \Lambda_j^\gamma)$ into $\text{Tr}[G_{\Lambda_{j,t}}^T d\Lambda_{j,t}]$, we get $\text{Tr}[G_{\Lambda_{j,t}}^T d\Lambda_{j,t}] = \text{Tr}[G_{\Lambda_j^\gamma}^T d\Lambda_j^\gamma] + \text{Tr}[G_{\phi_j}^T d\phi_j] + \text{Tr}[G_{D_j}^T dD_j]$. If one $\lambda_j^{(i)} \rightarrow 0$ in $\Lambda_j^\gamma = diag[\lambda_j^{(1)}, \dots, \lambda_j^{(m_j)}]$, discard the dimension $y_j^{(i)}$ and its corresponding subset of parameters in Θ_j .

As shown in Figure 5(C), the third way is embedding a temporal structure $q(y_t | \omega_t)$ into the distribution of $y_t^{(i)}$ in each subspace, via a regression parameterization $\omega_t = f(\sum_{\tau} \vartheta_{i,\tau} y_{t-\tau})$ on past samples of $y_t^{(i)}$ or estimating $q(y_t^{(i)})$ as a marginal distribution $q(y_t) = \sum_{y_{t-1}} q(y_t | y_{t-1}) q(y_{t-1})$ or $q(y_t) = \int q(y_t | y_{t-1}) q(y_{t-1}) dy_{t-1}$ via the distributions of past samples. Shown in Figure 8 are detailed equations. For subspaces of Gaussian $y_t^{(i)}$, $y_t = B_j y_{t-1} + e_t^\gamma$ is embedded into subspaces of local factor analysis, and thus has been studied under the name of local temporal factor analysis (TFA) (see Sec. IV, Xu, 2004).

Traced back to the early 1960's in the literature of nonparametric statistics, studies on RBF networks started from simple kernels $k(x, x_i)$ located at each sample and combined by linear weights that are simply samples of z . Then, studies proceeded along a direction not only with $k(x, x_i)$ extended to learning various structures in different subspaces and for different temporal dependences, but also with those combining weights estimated from samples or further extended to a learning gating structure.

Interestingly, a reversed direction of trends has also become popularized in recent years. One example is that SBF seeks to use simple linear subspaces instead of experts in a sophisticated structure. Another example is the widely studied support vector machine (SVM). It returns to considering locating each base or kernel $k(x, x_i)$ at each sample x_i . In help of convex optimization techniques (Rockafellar, 1972; Vapnik, 1995), learning is made both on weighting parameters for linear combination and on selecting a small subset of samples for locating kernels $k(x, x_i)$. In the past decade, efforts are made not only beyond the classic SVM limitation of only considering two category classification, but also on how to estimate an appropriate structure for kernels $k(x, x_i)$.

Cross-fertilization of two directions deserves be explored too. The first direction aims at modeling samples by seeking its distribution or structure, based on which classification and also other problem solving tasks are handled. While a crucial nature of SVM is seeking a discriminating strip such that two classes become more apart from each other, merely based on samples around the boundary of two classes. To classifying samples with a more sophisticated structure, studies on the second direction further proceed to estimate an appropriate structure for kernels, for which it is helpful to recall RBF studies that have experienced a path from simple kernels to various structures. On the other hand, studies of the first direction may also get a help from studies of the second direction by enhancing efforts on seeking a more robust boundary structure, e.g., in Algorithms II or III, with $\ell^* = \arg \max_{\ell} q(z = \ell | x, \theta_j)$ we require $q(z = \ell^* | x, \theta_j) - \max_{j \neq \ell^*} q(z = j | x, \theta_j)$ to be larger than a pre-specified threshold.

The last but not the least, its mathematical relation of Bayesian Ying Yang learning to the classical generalization error bound still remains an open issue. We recall that the concept of generalization error comes from measuring the discrepancy of a model estimated on a training set $\mathbf{X}_N = \{x_t\}_{t=1}^N$ from the true regularity of samples underlying \mathbf{X}_N . Actually this concept involves a truth-learner philosophy. That is, there are a truth or regularity and a learner. The learner can not directly see the truth or regularity but can get a set \mathbf{X}_N of samples from the truth or regularity subject to some disturbances or noises. The learner attempts to describe this \mathbf{X}_N as well as future samples from the same truth or regularity. It describes \mathbf{X}_N via measuring an error that the learner fits \mathbf{X}_N (e.g., those minimum fitting error based approaches) or how likely \mathbf{X}_N really comes from a model described by the learner (e.g., the maximum likelihood based methods). More generally, a generalization error attempts to measure an error that the learner fits not only \mathbf{X}_N but also any samples from the same truth or regularity, which has a prediction nature and thus is difficult to obtain accurately.

In a contrast, a Ying Yang harmony involves a relative philosophy about two matters or systems that interact each other, while the concept of learner-seeking- truth is extended to a concept about how close the two systems are, which may be observed from two general aspects. One is externally observing how the two systems describe \mathbf{X}_N , which leads to either a correlation type concept if we are interested in whether two descriptions share certain common points or an indifference concept (a relaxed version of equivalence concept) if we are further interested in how close or how different the two descriptions are. This scenario includes the truth-learner type as a special case that a system is simply the one that generates \mathbf{X}_N . Still, we may need to consider the concepts of correlation and indifference about how the

two systems describe samples beyond \mathbf{X}_N but from the same truth or regularity. This lacks study yet but likely involves a trading-off between a minimum fitting error and a least model complexity.

The other aspect is observing how the two systems are, both externally on describing \mathbf{X}_N and internally on the inner representation \mathbf{R} . Not only concepts of correlation and matching should be considered for two systems with respect to both \mathbf{X} and \mathbf{R} , but also each system $M(\mathbf{X}, \mathbf{R})$ may have two different architectures, e.g., we get two complement systems $p(\mathbf{X} | \mathbf{R})p(\mathbf{R})$ as Ying and $p(\mathbf{R} | \mathbf{X})p(\mathbf{X})$ as Yang for a joint distribution $p(\mathbf{X}, \mathbf{R})$, or generally $M(\mathbf{X} | \mathbf{R})M(\mathbf{R})$ and $M(\mathbf{R} | \mathbf{X})M(\mathbf{X})$ even beyond a probability theoretic framework. That is, we have the interaction between two complement systems, the one $M(\mathbf{X} | \mathbf{R})M(\mathbf{R})$ models or describes the external data \mathbf{X}_N , while the other $M(\mathbf{R} | \mathbf{X})M(\mathbf{X})$ consists and $M(\mathbf{X})$ that comes from the data \mathbf{X}_N directly or after smoothing of $M(\mathbf{R} | \mathbf{X})$ for perceiving \mathbf{X}_N . On such two complement systems that describe both an external data \mathbf{X}_N and the inner representation \mathbf{R} , a correlation type concept is further developed into a harmony concept under certain conservation principle (e.g. $\int p(\mathbf{X}, \mathbf{R})d\mathbf{X}d\mathbf{R} = 1$), which combines the concepts of a minimum fitting error and a least model complexity while not facing the difficulty of seeking an appropriate trade-off. In other word, this \mathbf{X}_N based Ying Yang harmony closely relates to the classical concept of generalization error that bases on future samples with a difficult prediction nature.

CONCLUSION

Studies on RBF networks have been reviewed along the streams of its developments in past decades. Backtracked to the early 1960's in the literature of nonparametric statistics on Parzen Window estimator and subsequently on kernel regression estimator, advances are featured not only by the era from nonparametric based simple kernels to parameter estimation based normalized RBF networks, but also by the era of extending simple kernels into certain structures, from studies on mixture of experts and its alternatives to studies on subspace based functions (SBF), as well as further extensions for exploring temporal structures among samples. Moreover, different types of typical learning algorithms have also been summarized under the Bayesian Ying Yang learning framework for learning not only normalized RBF, ME and alternatives, but also SBF and temporal extensions.

ACKNOWLEDGMENT

The work described in this paper was supported by a grant from the Research Grant Council of the Hong Kong SAR (Project No: CUHK4173/06E), and also supported by the program of Chang Jiang Scholars, Chinese Ministry of Education for Chang Jiang Chair Professorship in Peking University.

REFERENCES

- Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19, 714–723.

- Akaike, H. (1981). Likelihood of a model and information criteria. *Journal of Econometrics*, 16, 3–14.
- Amari, S. I., Cichocki, A., & Yang, H. (1996), A new learning algorithm for blind separation of sources, In Touretzky, Mozer, & Hasselmo (Eds.), *Advances in Neural Information Processing System 8*, MIT Press, 757-763.
- Bell, A., & Sejnowski, T. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7, 1129–1159.
- Botros, S. M., & Atkeson, C. G. (1991), Generalization properties of radial basis function, In Lippmann, Moody, & Touretzky (eds), *Advances in Neural Information Processing System 3*, Morgan Kaufmann Pub., 707-713.
- Bozdogan, H. (1987). Model Selection and Akaike's Information Criterion: The general theory and its analytical extension. *Psychometrika*, 52, 345–370.
- Broomhead, D. S., & Lowe, D. (1988). Multivariable functional interpolation and adaptive networks. *Complex Systems*, 2, 321–323.
- Cavanaugh, J. (1997). Unifying the derivations for the Akaike and corrected Akaike information criteria. *Statistics & Probability Letters*, 33, 201–208.
- Chang, P. R., & Yang, W. H. (1997). Environment-adaptation mobile radio propagation prediction using radial basis function neural networks. *IEEE Transactions on Vehicular Technology*, 46, 155–160.
- Chen, S., Cowan, C. N., & Grant, P. M. (1991). Orthogonal least squares learning algorithm for Radial basis function networks. *IEEE Transactions on Neural Networks*, 2, 302–309.
- Dayan, P., Hinton, G. E., Neal, R. M., & Zemel, R. S. (1995). The Helmholtz machine. *Neural Computation*, 7(5), 889–904.
- Devroye, L. (1981). On the almost everywhere convergence of nonparametric regression function estimates. *Annals of Statistics*, 9, 1310–1319.
- Devroye, L. (1987). *A Course in Density Estimation*. Boston: Birkhauser.
- Er, M. J. (2002). Face recognition with radial basis function (RBF) neural networks. *IEEE Transactions on Neural Networks*, 13(3), 697–710.
- Fang, S. C., Gao, D. Y., Shue, R. L., & Wu, S. Y. (2008). Canonical dual approach for solving 0-1 quadratic programming problems. *Journal of Industrial and Management Optimization*, 4(1), 125–142.
- Ghahramani, Z., & Beal, M. J. (2000). Variational inference for Bayesian mixtures of factor analysers. In Solla, Leen, & Muller (eds), *Advances in Neural Information Processing Systems 12*, MIT Press, 449-455.
- Girosi, F., & Poggio, T. (1990). Networks and the best approximation property. *Biological Cybernetics*, 63(3), 169–176.

- Guerra, F. A., & Coelho, L. S. (2008). Multi-step ahead nonlinear identification of Lorenz's chaotic system using radial basis neural network with learning by clustering and particle swarm optimization. *Chaos, Solitons, and Fractals*, 35(5), 967–979.
- Hartman, E. J., Keeler, J. D., & Kowalski, J. M. (1990). Layered neural networks with Gaussian hidden units as universal approximations. *Neural Computation*, 2, 210–215.
- Heinen, T. (1996). *Latent class and discrete latent trait models: Similarities and differences*. Housand Oaks, California: Sage.
- Hinton, G. E., Dayan, P., Frey, B. J., & Neal, R. N. (1995). The wake-sleep algorithm for unsupervised learning neural networks. *Science*, 268, 1158–1160.
- Hinton, G. E., & Zemel, R. S. (1994), Autoencoders, minimum description length and Helmholtz free energy, In Cowan, Tesauro, & Alspector (eds), *Advances in Neural Information Processing Systems 6*, Morgan Kaufmann Pub., 449-455.
- Isaksson, M., Wisell, D., & Ronnow, D. (2005). Wide-band dynamic modeling of power amplifiers using radial-basis function neural networks. *IEEE Transactions on Microwave Theory and Techniques*, 53(11), 3422–3428.
- Jaakkola, T. S. (2001), Tutorial on variational approximation methods, in Opper & Saad (eds), *Advanced Mean Field Methods: Theory and Practice*, MIT press, 129-160.
- Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. *Neural Computation*, 3, 79–87.
- Jordan, M., Ghahramani, Z., Jaakkola, T., & Saul, L. (1999). Introduction to variational methods for graphical models. *Machine Learning*, 37, 183–233.
- Jordan, M. I., & Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6, 181–214.
- Jordan, M. I., & Xu, L. (1995). Convergence Results for The EM Approach to Mixtures of Experts Architectures. *Neural Networks*, 8, 1409–1431.
- Karami, A., & Mohammadi, M. S. (2008). Radial basis function neural network for power system load-flow. *International Journal of Electrical Power & Energy Systems*, 30(1), 60–66.
- Kardirkamanathan, V., Niranjan, M., & Fallside, F. (1991), Sequential adaptation of Radial basis function neural networks and its application to time-series prediction, In Lippmann, Moody, & Touretzky (eds), *Advances in Neural Information Processing System 3*, Morgan Kaufmann Pub., 721-727.
- Konishi, S., Ando, T., & Imoto, S. (2004). Bayesian information criteria and smoothing parameter selection in radial basis function networks. *Biometrika*, 91(1), 27–43.
- Lee, J. (1999). A practical radial basis function equalizer. *IEEE Transactions on Neural Networks*, 10, 450–455.
- Lin, G. F., & Chen, L. H. (2004). A non-linear rainfall-runoff model using radial basis function network. *Journal of Hydrology (Amsterdam)*, 289, 1–8.

- Ma, S., Ji, C., & Farmer, J. (1997). An Efficient EM-based Training Algorithm for Feedforward Neural Networks. *Neural Networks*, 10(2), 243–256.
- MacKay, D. (2003), Information Theory, Inference, and Learning Algorithms, Cambridge University Press.
- Mackey, D. (1992). A practical Bayesian framework for backpropagation. *Neural Computation*, 4, 448–472.
- Mai-Duy, N., & Tran-Cong, T. (2001). Numerical solution of differential equations using multiquadric radial basis function networks . *Neural Networks*, 14(2), 185–199.
- McLachlan, G. J., & Geoffrey, J. (1997), The EM Algorithms and Extensions, Wiley.
- Mel, B. W., & Omohundro, S. M. (1991), How receptive field parameters affect neural learning. In Lippmann, Moody, & Touretzky (eds), Advances in Neural Information Processing System 3, Morgan Kaufmann Pub., 757-763.
- Moody, J., & Darken, C. (1989). Fast learning in networks of locally-tuned processing units . *Neural Computation*, 1, 281–294.
- Neath, A. A., & Cavanaugh, J. E. (1997). Regression and Time Series model selection using variants of the Schwarz information criterion. *Communications in Statistics A*, 26, 559–580.
- Nowlan, S. J. (1990), Max likelihood competition in RBF networks, TR. CRG-Tr-90-2, U. of Toronto, Dept. of Computer Science.
- Park, J., & Sandberg, I. W. (1993). Universal approximation using radial-basis-function networks. *Neural Computation*, 5, 305–316.
- Poggio, T., & Girosi, F. (1990). Networks for approximation and learning, Networks for approximation and learning. *Proceedings of the IEEE*, 78, 1481–1497.
- Powell, M. J. D. (1987), Radial basis functions for multivariable interpolation: a review, in Mason & Cox (Eds.), Algorithms for Approximation, Oxford: Clarendon Press.
- Reddy, R., & Ganguli, R. (2003). Structural damage detection in a helicopter rotor blade using radial basis function neural networks. *Smart Materials and Structures*, 12, 232–241.
- Redner, R. A., & Walker, H. F. (1984). Mixture densities, maximum likelihood, and the EM algorithm. *SIAM Review*, 26, 195–239.
- Rissanen, J. (1986). Stochastic complexity and modeling. *Annals of Statistics*, 14(3), 1080–1100.
- Rissanen, J. (1989), Stochastic Complexity in Statistical Inquiry, World Scientific: Singapore.
- Rockafellar, R. (1972), Convex Analysis, Princeton University Press.
- Rumelhart, D. E., Hintont, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533–536.

- Salah, A. A., & Alpaydin, E. (2004), Incremental mixtures of factor analyzers, Proc. the 17th International Conference on Pattern Recognition, 23-26 Aug. 2004, Cambridge, UK, Vol.1, 276-279.
- Sarimveis, H., Doganis, P., & Alexandridis, A. (2006). classification technique based on radial basis function neural networks. *Advances in Engineering Software*, 37(4), 218–221.
- Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6, 461–464.
- Shi, L. (2008), Bayesian Ying-Yang harmony learning for local factor analysis: a comparative investigation, In Tizhoosh & Ventresca (eds), Oppositional Concepts in Computational Intelligence, Springer-Verlag, 209-232.
- Specht, D. F. (1990). Probabilistic neural networks. *Neural Networks*, 3, 109–118.
- Stone, M. (1978). Cross-validation: A review. *Math. Operat. Statist.*, 9, 127–140.
- Sun, K., Tu, S. K., Gao, D. Y., & Xu, L. (2009), Canonical Dual Approach to Binary Factor Analysis. In T. Adali et al. (Eds.), ICA 2009 (LNCS 5441), pp 346-353.
- Treier, S., & Jackman, S. (2002), Beyond factor analysis: modern tools for social measurement. Presented at the 2002 Annual Meetings of the Western Political Science Association and the Midwest Political Science Association.
- Vapnik, V. (1995), The Nature Of Statistical Learning Theory, Springer.
- Vapnik, V. (2006), Estimation of Dependences Based on Empirical Data, Springer.
- Wallace, C., & Freeman, P. (1987). Estimation and inference by compact coding. *Journal of the Royal Statistical Society. Series A (General)*, 49(3), 240–265.
- Wallace, C. S., & Boulton, D. M. (1968). An information measure for classification. *The Computer Journal*, 11, 185–194.
- Xu, L. (1995), Bayesian-Kullback coupled YING-YANG machines: unified learning and new results on vector quantization, Proc.ICONIP95, Oct 30-Nov.3, 1995, Beijing, pp 977-988. A further version in NIPS8, D.S. Touretzky, et al (Eds.), MIT Press, 444–450.
- Xu, L. (1998). RBF nets, mixture experts, and Bayesian Ying-Yang learning. *Neurocomputing*, 19(1-3), 223–257.
- Xu, L. (2001). Best Harmony, Unified RPCL and Automated Model Selection for Unsupervised and Supervised Learning on Gaussian Mixtures, ME-RBF Models and Three-Layer Nets. *International Journal of Neural Systems*, 11(1), 3–69.
- Xu, L. (2001). BYY harmony learning, independent state space and generalized APT financial analyses. *IEEE Transactions on Neural Networks*, 12(4), 822–849.
- Xu, L. (2002). BYY harmony learning, structural RPCL, and topological self-organizing on unsupervised and supervised mixture models. *Neural Networks*, 15, 1125–1151.

- Xu, L. (2004a). Advances on BYY harmony learning: information theoretic perspective, generalized projection geometry, and independent factor auto-determination. *IEEE Transactions on Neural Networks*, 15, 885–902.
- Xu, L. (2004b). Temporal BYY encoding, Markovian state spaces, and space dimension determination. *IEEE Transactions on Neural Networks*, 15, 1276–1295.
- Xu, L (2005), Fundamentals, Challenges, and Advances of Statistical Learning for Knowledge Discovery and Problem Solving: A BYY Harmony Perspective, Keynote talk. *Proc. Of Intl. Conf. on Neural Networks and Brain*, Oct. 13-15, 2005, Beijing, China, Vol. 1, 24-55.
- Xu, L. (2007a), Bayesian Ying Yang Learning, Scholarpedia 2(3):1809, Retrieved from http://scholarpedia.org/article/Bayesian_Ying_Yang_learning.
- Xu, L. (2007b), Rival penalized competitive learning, Scholarpedia 2(8):1810, Retrieved from http://www.scholarpedia.org/article/Rival_penalized_competitive_learning
- Xu, L. (2007c), A trend on regularization and model selection in statistical learning: a Bayesian Ying Yang learning perspective, In Duch & Mandziuk (eds.), *Challenges for Computational Intelligence*, Springer-Verlag, 365-406.
- Xu, L. (2007d). A unified perspective and new results on RHT computing, mixture based learning, and multi-learner based problem solving. *Pattern Recognition*, 40, 2129–2153.
- Xu, L. (2008a), Bayesian Ying Yang System, Best Harmony Learning, and Gaussian Manifold Based Family, In Zurada et al (eds.) Computational Intelligence: Research Frontiers, WCCI2008 Plenary/ Invited Lectures, LNCS5050, 48–78.
- Xu, L. (2008b). (in press). Machine learning problems from optimization perspective, A special issue for CDGO 07. *Journal of Global Optimization*.
- Xu, L. (2008c), Independent Subspaces, in Ramón, Dopico, Dorado & Pazos (Eds.), *Encyclopedia of Artificial Intelligence*, IGI Global (IGI) publishing company, 903-912.
- Xu, L., Jordan, M. I., & Hinton, G. E. (1994), A Modified Gating Network for the Mixtures of Experts Architecture, Proc. of WCNN94, San Diego, CA, 405-410.
- Xu, L., Jordan, M. I., & Hinton, G. E. (1995), An Alternative Model for Mixtures of Experts, In Tesauro, Touretzky & Leen (eds), *Advances in Neural Information Processing Systems 7*, MIT Press, 633-640.
- Xu, L., Krzyzak, A., & Oja, E. (1993). Rival Penalized Competitive Learning for Clustering Analysis, RBF net and Curve Detection. *IEEE Trans. Neural Networks*, 4(4), 636-649. An early version on Proc. 1992 IJCNN, Nov.3-6, 1992, Beijing, 665-670.
- Xu, L., Krzyzak, A., & Yuille, A. L. (1994). On Radial Basis Function Nets and Kernel Regression: Statistical Consistency, Convergence Rates and Receptive Field Size. *Neural Networks*, 7(4), 609–628.
- Yuille, A. L., & Grzywacz, N. M. (1989). A mathematical analysis of the motion coherence theory. *International Journal of Computer Vision*, 3, 155–175.

KEY TERMS AND THEIR DEFINITIONS

Normalized Radial Basis Function (NRBF) and Extended NRBF: A radial basis function (RBF) $f(x) = \sum_{j=1}^k w_j \varphi_j(x - c_j, \theta_j)$ is a linear combination of a series of simple function $\{\varphi_j(x - c_j, \theta_j)\}_{j=1}^k$, with each called a base that is located at c_j . Each base is classically radial symmetrical from its center c_j , while it becomes unnecessary presently. A RBF is called normalized RBF (NRBF) if we have $\sum_{j=1}^k \varphi_j(x - c_j, \theta_j) = 1$, and is further called Extended NRBF when each constant w_j is extended into a function $f_j(x)$. Typically, we consider the cases that $f_j(x) = W_j x + w_j$ is a linear function.

Mixtures-of-Experts (ME) and Alternative ME (AME): Initially, a mixture-of-experts is a weighted combination $f(x) = \sum_{j=1}^k g_j(x, \phi) f_j(x, \theta_j)$ of a number of functions with each $f_j(x, \theta_j)$ called expert that is weighted by $g_j(x, \phi) = e^{v_j(x, \phi)} / \sum_{j=1}^k e^{v_j(x, \phi)}$ that is called a gating net with each $v_j(x, \phi)$ implemented by a three layer forward net. Generally, this weighted combination is actually the regression equation of a mixture of conditional distributions, i.e., $q(z | x) = \sum_{j=1}^k g_j(x, \phi) q(z | x, \theta_j)$. All the unknowns are estimated via the maximum likelihood (ML) learning on $q(z | x)$ by a generalized Expectation and Maximization (EM) algorithm. Moreover, alternative ME is a particular ME family with $p(z | x)$ supported on a finite mixture $q(x | \phi) = \sum_{j=1}^k \alpha_j q(x | \phi_j)$, $\sum_{j=1}^k \alpha_j = 1$, $\alpha_j \geq 0$, and with its corresponding posteriori as the gating net $g_j(x, \phi) = \alpha_j q(x | \phi_j) / q(x | \phi)$. All the unknowns are estimated via the ML learning on $q(z | x) q(x | \phi) = \sum_{j=1}^k \alpha_j q(x | \phi_j) q(z | x, \theta_j)$ by the EM algorithm.

Subspace Based Functions and Cascaded Extensions: Considering $q(x | \phi_j)$ as generated from a subspace with independent factors distributed along each coordinate of an inner m dimensional representation $y = [y^{(1)}, \dots, y^{(m)}]$, we get basis functions or the gating net by $g_j(x, \phi) = \alpha_j q(x | \phi_j) / q(x | \phi)$ based on different subspaces, resulting in subspace based extensions of ME and RBF networks. That is, we get $f(x) = \sum_{j=1}^k g_j(x, \phi) f_j(x, \theta_j)$ as a combination of functions supported on subspace bases, which is thus called subspace based functions (SBF). Moreover, a direct mapping $x \rightarrow z$ by $f_j(x, \theta_j)$ can be replaced by a mapping $x \rightarrow y$ as feature extraction or dimension reduction and then followed by another mapping $y \rightarrow z$, from which we get a cascade $x \rightarrow y \rightarrow z$ mapping. This cascaded extension will discard redundant parts with further improvements on performance.

Model Selection and Two Stage Implementation: It refers to select an appropriate one among a family of infinite many candidate structures $\{S_k(\Theta_k)\}$ with each $S_k(\Theta_k)$ in a same configuration but in different scales, each of which is labeled by a scale parameter k in term of one integer or a set of integers. Selecting an appropriate k means getting a structure consisting of an appropriate number of free parameters. Usually, a maximum likelihood (ML) learning is not good for model selection. Classically, model selection is made in a two stage implementation. First, enumerate a candidate set K of k and estimate the unknown set Θ_k of parameters by ML learning for a solution Θ_k^* at each $k \in K$. Second, use a model selection criterion $J(\Theta_k^*)$ to select a best k^* . Several criteria are available for the purpose, such as AIC, CAIC, BIC, cross validation, etc.

Rival Penalized Competitive Learning (RPCL): It is a further development of competitive learning in help of an appropriate balance between participating and leaving mechanisms, such that an appropriate number k of individual substructures will be allocated to learn multiple structures underlying observations. With k initially at a value larger enough, the participating mechanism is featured by that

a coming sample x_t is allocated to one of the k substructures via competition, and the winner adapts this sample by a little bit, while the leaving mechanism is featured by that the rival is de-learned a little bit to reduce a duplicated allocation, which will discard extra substructures, with model selection made automatically during learning.

Bayesian Ying-Yang System: A set $\mathbf{X} = \{x\}$ of samples and its inner representation \mathbf{R} in an intelligent system are jointly considered by their joint distribution in two types of Bayesian decomposition. In a compliment to the famous ancient Ying-Yang philosophy, one decomposition $p(\mathbf{X}, \mathbf{R}) = p(\mathbf{R} | \mathbf{X})p(\mathbf{X})$ coincides the Yang concept with a visible domain $p(\mathbf{X})$ as a Yang space and a forward pathway by $p(\mathbf{R} | \mathbf{X})$ as a Yang pathway. Thus, $p(\mathbf{X}, \mathbf{R})$ is called Yang machine. Also, $q(\mathbf{X}, \mathbf{R}) = q(\mathbf{X} | \mathbf{R})q(\mathbf{R})$ is called Ying machine with an invisible domain $q(\mathbf{R})$ as a Ying space and a backward pathway by $q(\mathbf{X} | \mathbf{R})$ as a Ying pathway. Such a Ying-Yang pair is called Bayesian Ying-Yang system. The input to the Ying Yang system is through $p(\mathbf{X}) = p(\mathbf{X} | \mathbf{X}_N, h)$ directly from a training sample set $\mathbf{X}_N = \{x_t\}_{t=1}^N$, while the inner representation $q(\mathbf{R}) = q(\mathbf{Y}, \Theta) = q(\mathbf{Y} | \Theta)q(\Theta | \Xi)$ describes both a long term memory Θ that is a collection of all unknown parameters in the system and a short term memory \mathbf{Y} with each $y \in \mathbf{Y}$ corresponding to one element $x \in X$. To build up an entire system, we need to design appropriate structures for each component. Specifically, the structure of $q(\mathbf{Y} | \Theta)$ is designed subject to the nature of learning tasks and a principle of least representation redundancy, the structure of $q(\mathbf{X} | \mathbf{R})$ is designed to suit the mapping $\mathbf{Y} \rightarrow \mathbf{X}$ under a principle of divide and conquer so that a complicated mapping is realized by a number of simple ones, while the structure of $p(\mathbf{R} | \mathbf{X})$ is designed for an inverse map $\mathbf{X} \rightarrow \mathbf{Y}$ under a principle of uncertainty conversation between Ying-Yang, i.e., Yang machine preserves a room or varying range that is appropriate to accommodate uncertainty or information contained in the Ying machine.

Bayesian Ying Yang Learning: Named in a compliment to the famous ancient Chinese Ying-Yang philosophy, it refers to a general statistical learning framework that formularizes learning tasks in a two pathway featured intelligent system via two complementary Bayesian representations of the joint distribution on the external observation and its inner representation, with all unknowns in the system determined by a principle that two Bayesian representations become best harmony. This system is called Bayesian Ying Yang system, mathematically described by $q(\mathbf{X}, \mathbf{R}) = q(\mathbf{X} | \mathbf{R})q(\mathbf{R})$ and $p(\mathbf{X}, \mathbf{R}) = p(\mathbf{R} | \mathbf{X})p(\mathbf{X})$. This best harmony is mathematically implemented by maximizing $H(p || q) = \int p(\mathbf{R} | \mathbf{X})p(\mathbf{X}) \ln[q(\mathbf{X} | \mathbf{R})q(\mathbf{R})]d\mathbf{X}d\mathbf{R}$, called Bayesian Ying Yang harmony learning. It follows from $H(p || q) = -KL(p || q) + \int p(\mathbf{R} | \mathbf{X})p(\mathbf{X}) \ln[p(\mathbf{R} | \mathbf{X})p(\mathbf{X})]d\mathbf{X}d\mathbf{R}$ that this best Ying Yang harmony principle includes not only a best Ying Yang matching by minimizing the Ying Yang divergence $KL(p || q) = \int p(\mathbf{R} | \mathbf{X})p(\mathbf{X}) \ln[p(\mathbf{R} | \mathbf{X})p(\mathbf{X}) / q(\mathbf{X} | \mathbf{R})q(\mathbf{R})]d\mathbf{X}d\mathbf{R}$, but also minimizing the entropy $-\int p(\mathbf{R} | \mathbf{X})p(\mathbf{X}) \ln[p(\mathbf{R} | \mathbf{X})p(\mathbf{X})]d\mathbf{X}d\mathbf{R}$ of the Yang machine. In other words, a best Ying Yang harmony seeks a Yang machine as an inverse of Ying machine such that it best matches the Ying machine and also keeps a least complexity. Moreover, this best Ying Yang matching provides a general perspective that unifies a number of typical statistical learning approaches.

Automatic Model Selection: Being different from a usual incremental or decremental model selection that bases on evaluating the change $J(\Theta_k) - J(\Theta_k \cup \theta_{new})$ as a subset θ_{new} of parameters is added or removed, automatic model selection is associated with not only a learning algorithm or a learning principle but also an indicator $\rho(\theta_r)$ on a subset $\theta_r \in \Theta_k$. If θ_r consists of parameters of a redundant structural part, learning via either implementing this learning algorithm or optimizing this learning principle will

drive $\rho(\theta_r) \rightarrow 0$ and θ_r towards a specific value, such that the corresponding redundant structural part is effectively removed. One example of such a learning algorithm is Rival Penalized Competitive Learning, while one example of such a learning principle is Bayesian Ying-Yang Harmony Learning.

Chapter 4

Nature Inspired Methods for Multi-Objective Optimization

Sanjoy Das

Kansas State University, USA

Bijaya K. Panigrahi

Indian Institute of Technology, India

Shyam S. Pattnaik

National Institute of Technical Teachers' Training & Research, India

ABSTRACT

This chapter focuses on the concepts of dominance and Pareto-optimality. It then addresses key issues in applying three basic classes of nature inspired algorithms – evolutionary algorithms, particle swarm optimization, and artificial immune systems, to multi-objective optimization problems. As case studies, the most significant multi-objective algorithm from each class is described in detail. Two of these, NSGA-II and MOPSO, are widely used in engineering optimization, while the others show excellent performances. As hybrid algorithms are becoming increasingly popular in optimization, this chapter includes a brief discussion of hybridization within a multi-objective framework.

INTRODUCTION

Many real world optimization problems cannot be formulated readily as one involving either the minimization or the maximization of a single objective function. Under these circumstances, the concepts of dominance and Pareto-optimality are usually invoked (Deb, 2001; Das & Panigrahi, 2008). A multi-objective framework provides a formal basis to develop effective optimization algorithms, and to evaluate their performances.

Nature inspired algorithms, heuristic approaches that tend to mimic various natural phenomena, have been very successful in addressing multi-objective optimization problems. These algorithms are typically population-based approaches that store a set (or population) of solutions, which are regularly updated – a feature that is significant in a multi-objective framework, as it is not easy to discern between

DOI: 10.4018/978-1-60566-766-9.ch004

a good solution and a bad one. Furthermore, being stochastic approaches these algorithms are equipped to handle local minima in the fitness landscape. Lastly, these methods can be hybridized readily with greedy algorithms for faster convergence (Das, 2008).

Evolutionary algorithms are a popular scheme for nature-inspired multi-objective optimization (Goldberg, 1989; Mitchell, 1998). These algorithms are based on Darwinian mechanisms of natural selection. A recent nature inspired approach is particle swarm optimization (PSO). PSO borrows from swarm intelligence, where simple interactions between individuals (called particles) enable the swarm to converge to optimal locations in the search space (Clerc, 2005). Yet another widely used widely used natural paradigm for multi-objective optimization is based on artificial immune systems (AIS) (de Castro & Von Zuben, 2002).

The multi-objective optimization has been applied to train a wide variety of machine learning models. Usually, in these cases, the training error is treated as only one of the objectives for optimization, while other aspects of the learning system, such as regularization features, network size, or number of kernels, are also considered as other objectives.

It must be noted here that this chapter deals with continuous optimization problems. Combinatorial optimization problems are another class of problems, for which multi-objective algorithms exist (*e.g.* multi-objective ant colony optimization). However, such studies are not as extensive as in case of continuous optimization. Moreover, the usefulness of combinatorial optimization algorithms in machine learning is rather limited. Therefore, combinatorial optimization shall not be addressed in this chapter. Needless to say, the nature inspired approaches discussed here can be modified for combinatorial optimization problems.

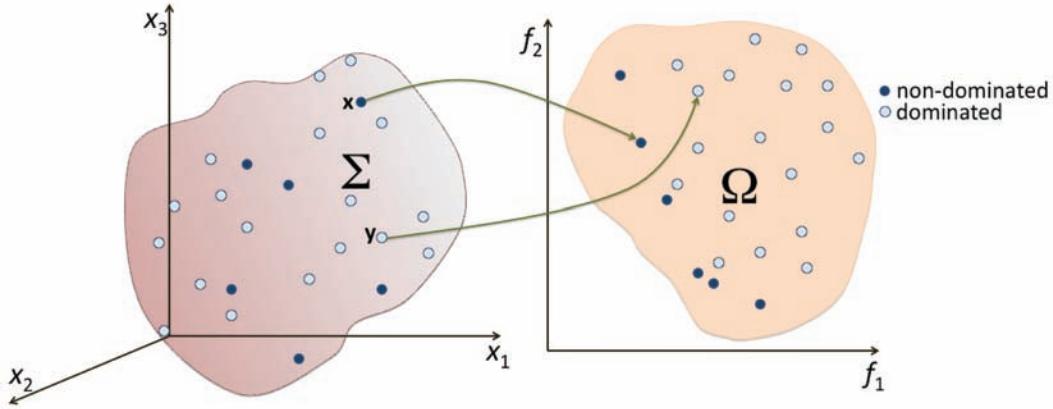
BASIC CONCEPTS OF MULTI-OBJECTIVE OPTIMIZATION

Let the search space of the multi-objective problem (*i.e.* the space of all solutions) be denoted as $\Sigma \subseteq \Re^N$, where N is the dimension of each solution vector. Without loss of generality, it shall be assumed that the multi-objective optimization problem involves the simultaneous minimization of M objectives, f_1, f_2, \dots, f_M . The objective function space $\Omega \subseteq \Re^M$ is defined as the set $\{(f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_M(\mathbf{x})) \mid \mathbf{x} \in \Sigma\}$. It should be noted that typically, the dimensionality of the objective function space, M , is much lower than that of the search space, N . The relationship between the search space and the objective function space is illustrated in Figure 1. Below, for a simple case where $N = 3$ and $M = 2$.

In the same figure are also shown two solutions, \mathbf{x} and \mathbf{y} in Σ , as well as their images in Ω . From the latter, it is clear that \mathbf{x} is a better solution than \mathbf{y} , as $f_1(\mathbf{x}) < f_1(\mathbf{y})$ and $f_2(\mathbf{x}) < f_2(\mathbf{y})$. We say that \mathbf{x} dominates \mathbf{y} . Mathematically this relationship is expressed as $\mathbf{x} \prec \mathbf{y}$. When M objectives are involved, we say that a solution, \mathbf{x} dominates another one, \mathbf{y} if and only if $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$ for $i = 1, 2, \dots, M$, with the inequality being strict for at least one objective, *i.e.*, $f_i(\mathbf{x}) < f_i(\mathbf{y})$ for some i . In any population of solutions \mathbf{P} , the set of all solutions that are not dominated by any other solution is called the *non-dominated set* $\Phi(\mathbf{P})$. Mathematically we write,

$$\Phi(\mathbf{P}) = \{\mathbf{x} \in \mathbf{P} \mid \text{for all } \mathbf{y} \in \mathbf{P}, \mathbf{y} \prec \mathbf{x} \text{ is false}\}. \quad (1)$$

Figure 1. Relationship between the search space (Σ) and the objective function space (Ω) showing a finite set of dominated and non-dominated solutions.



The image of $\Phi(\mathbf{P})$ in Ω is called the *non-dominated front*. Figure 1. shows sample dominated and non-dominated solutions. The non-dominated set of the entire search space $\Phi(\Sigma)$ is called the *Pareto set*, and its image, the *Pareto front* $\Phi(\Omega)$.

The goal of multi-objective optimization is to obtain samples solutions that are as close as possible to the Pareto front. The proximity of the non-dominated solutions provided by an algorithm to the Pareto front reflects its performance. We shall refer to this feature as *convergence*. In addition to convergence to the Pareto front, the algorithm must also provide samples that cover as much of the latter as possible. This desirable feature we shall refer to as *diversity* (Das & Panigrahi, 2008).

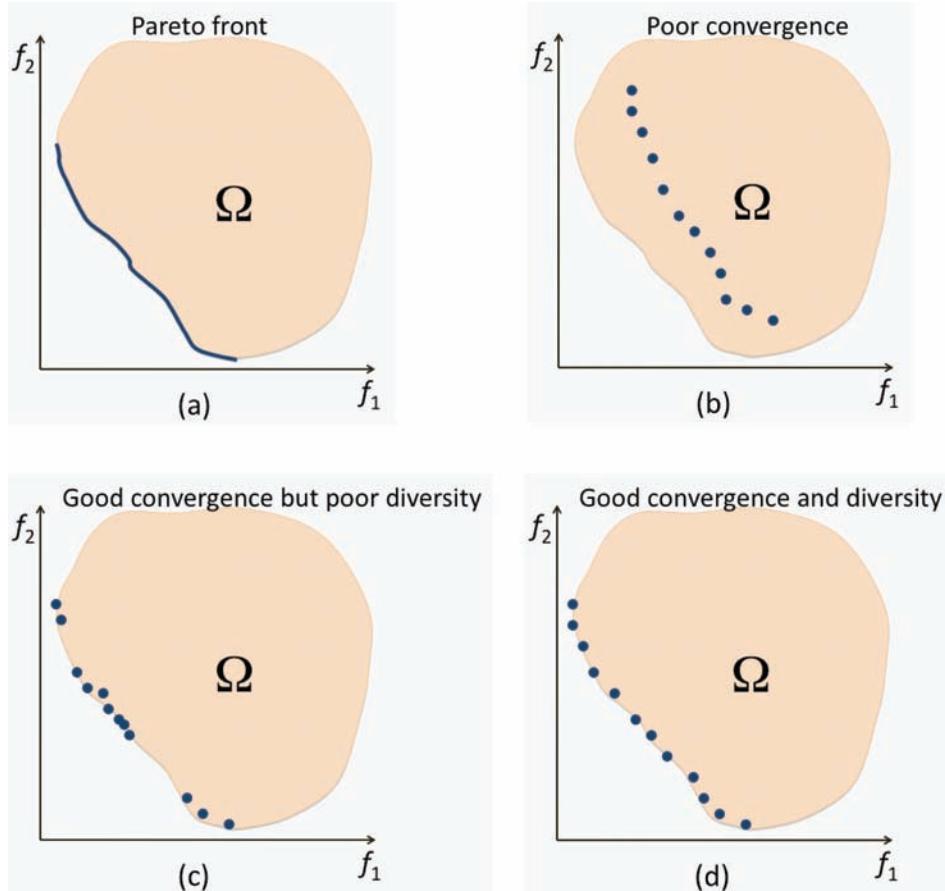
Figure 2 illustrates convergence and diversity. Figure 2 (a) shows the Pareto front $\Phi(\Omega)$. Figure 2 (b) shows sample solutions obtained from an algorithm, that are not close to it, indicating poor convergence towards the front. In Figure 2 (c), the solutions are in the front, but not regularly spaced, and regions of the front have been left out. Hence the non-dominated front lacks diversity. Lastly, Figure 2 (d) shows a set of sample solutions that are good in terms of both convergence and diversity.

EVOLUTIONARY ALGORITHMS

Evolutionary algorithms maintain a population of solutions of a fixed size P , where the solutions are called *chromosomes*. We shall denote the population during a generation t as \mathbf{P}_t . The initial population, \mathbf{P}_0 may be initialized randomly. In each generation t , the fitness of chromosomes in \mathbf{P}_t are evaluated, following which, they are subject to three evolutionary operators, *selection*, *crossover*, and *mutation* (Goldberg, 1989; Mitchell, 1998).

Selection is the process by which chromosomes are picked (with replacement) in such a manner that more fit ones have a higher probability of getting selected. In one such scheme called roulette wheel selection, the probability of selecting a chromosome \mathbf{x} from \mathbf{P}_t is,

Figure 2. Convergence and diversity issues.



$$p(\mathbf{x}) = \frac{f(\mathbf{x})}{\sum_{\mathbf{y} \in P_t} f(\mathbf{y})} \quad (2)$$

Another popular scheme is binary tournament selection. In this case, two chromosomes are picked randomly from P_t and a tournament is held between them to obtain the better one in terms of the fitness.

During crossover, two selected parent chromosomes are combined to obtain a new one, called the offspring. Crossover is designed in a manner that the offspring bears characteristics that are intermediate to the parents. For example, in convex crossover the offspring produced by two given parents, \mathbf{x} and \mathbf{y} , is given by,

$$\lambda\mathbf{x} + (1-\lambda)\mathbf{y}, \quad (3)$$

where λ is randomly generated such that $0 \leq \lambda \leq 1$. Another offspring may be created from the same set of parents, \mathbf{x} and \mathbf{y} by interchanging their positions in Eqn. 3.

The offspring are subjected to mutation, where a small perturbation is added to it. One common strategy is Gaussian mutation where the mutant of \mathbf{x} is given by,

$$\mathbf{x} = \mathbf{x} \pm N(0, \sigma). \quad (4)$$

The variance σ of the normal distribution N is kept at a very small value so that mutation does not result in changes that are too dramatic.

Using selection, crossover and mutation, a total of P offspring are obtained, who then replace the original population \mathbf{P} . In some versions a few of the fittest parents may be included in the new population for the next generation. This strategy, called elitism, guarantees that the best chromosomes of any generation are not lost, and is shown to produce faster convergence to the optima.

Multi-Objective Evolutionary Algorithms

Several evolutionary algorithms for multi-objective optimization have been proposed (Deb, 2001; Das and Panigrahi, 2008). However, since NSGA-II (Non Dominated Genetic Algorithm-II) is perhaps the classic approach for most applications, a more detailed description of this algorithm is provided here (Deb *et al.*, 2002). NSGA-II maintains two separate populations of parents \mathbf{P}_t and offspring \mathbf{Q}_t of equal size, P .

In each iteration t , the parent and offspring population from the previous iteration are merged together. The chromosomes in this merged population, $\mathbf{P}_{t-1} \cup \mathbf{Q}_{t-1}$, are then ranked using a ranking scheme called *non-dominated sorting*. Non-dominated sorting assigns ranks to each solution, and divides $\mathbf{P}_{t-1} \cup \mathbf{Q}_{t-1}$ into disjoint sets, \mathbf{F}_r , $r = 0, 1, \dots$ such that all solutions with any rank r are placed in the corresponding \mathbf{F}_r . It begins by assigning ranks of zero to the non-dominated set. In order to obtain every other set \mathbf{F}_r , $r > 0$, the previous sets \mathbf{F}_0 through \mathbf{F}_{r-1} are removed, and the non-dominated set of the remaining ones are placed in \mathbf{F}_r . In other words, \mathbf{F}_r can be recursively defined as,

$$\mathbf{F}_r = \begin{cases} \mathbf{F}(\mathbf{P}_{t-1} \cup \mathbf{Q}_{t-1}) & r = 0 \\ \mathbf{F}(\mathbf{P}_{t-1} \cup \mathbf{Q}_{t-1} - (\mathbf{F}_0 \cup \mathbf{F}_1 \cup \dots \cup \mathbf{F}_{r-1})) & r > 0 \end{cases} \quad (5)$$

Starting with \mathbf{F}_0 the lowest ranked chromosomes are inserted into an initially empty \mathbf{P}_t , the parent population of the current generation, until the size $|\mathbf{P}_t|$ reaches its limit, P .

When the last set, say \mathbf{F}_s , cannot be fully accommodated within \mathbf{P}_t (*i.e.* $\sum_{r=0}^{s-1} |\mathbf{F}_r| < P < \sum_{r=0}^s |\mathbf{F}_r|$), the diversity criterion is invoked to identify the best $P - \sum_{r=0}^{s-1} |\mathbf{F}_r|$ chromosomes. This is done using the *bounding hypercube* heuristic that surrounds each chromosome with the largest possible hypercube that includes no other solution except itself. To do so, sorted lists of the chromosomes in \mathbf{F}_s based on each objective are obtained. Suppose the immediate predecessor and successor of a chromosome \mathbf{x} in \mathbf{F}_s when using the sorted list of the m^{th} objective f_m , are denoted as \mathbf{x}_-^m and \mathbf{x}_+^m , the perimeter of this bounding hypercube is,

$$H(\mathbf{x}) = \sum_{m=1}^M |\mathbf{x}_+^m - \mathbf{x}_-^m| \quad (6)$$

Figure 3. The bounding hypercube and adaptive hypergrid methods for diversity.

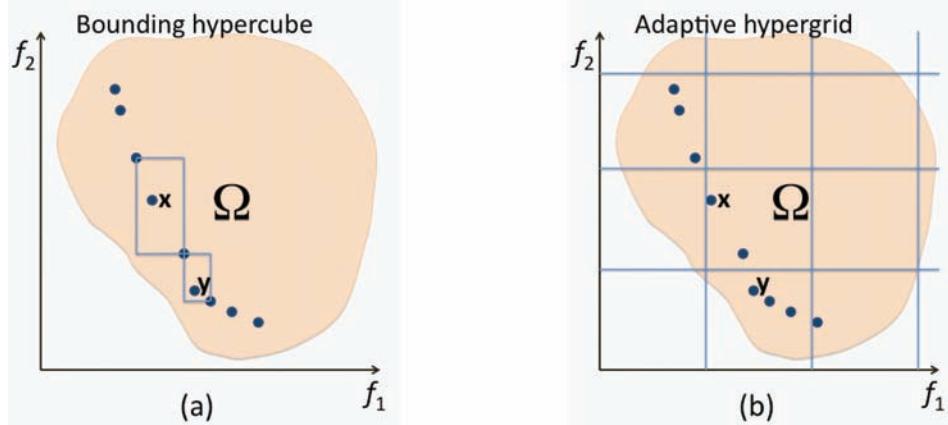


Figure 3 (a) shows how this scheme is implemented when two objectives are present. Between the two chromosomes \mathbf{x} and \mathbf{y} shown, \mathbf{x} is preferred as it is bounded by a larger hypercube than \mathbf{y} . Extreme chromosomes, which either have no predecessor or no successor in a sorted list, are assigned hypercube perimeters of infinity. This perimeter $H(\mathbf{x})$ serves as a rough measure of the proximity of \mathbf{x} to other chromosomes, with higher values indicating that the corresponding solution is in a sparser region. The chromosomes in \mathbf{F}_s are sorted again based on this measure and inserted into \mathbf{P}_r , starting with the highest one, until the limit is reached.

The offspring population \mathbf{Q}_t is obtained from \mathbf{P}_r by applying the usual evolutionary operators. Binary tournament selection is used, where the non-dominated sorting and bounding hypercube criteria are used to determine the fitter chromosome. Although many different crossover and mutation scheme can be used, NSGA-II usually incorporates simulated binary crossover and polynomial mutation (Deb *et al.* 2002).

There are several other recent multi-objective evolutionary algorithms. The Pareto Archive Evolutionary Strategy (PAES) is such a method that uses mutation, but no crossover (Knowles & Corne, 2000). Parallel Efficient Global Optimization (ParEGO) is a method that is designed for problems where the computational cost of evaluating the fitness of a solution is expensive.

PARTICLE SWARM OPTIMIZATION

PSO is a more recent method for optimization that is grounded on the collective behavior of a swarm of organisms (Clerc, 2005). As was the case with evolutionary algorithms, PSO maintains a population \mathbf{P}_t of solutions in each generation t . Each such solution, or *particle*, is moving inside the search space, Σ with a certain velocity. This velocity is constantly updated so that the particle's movement gets redirected towards more promising regions in Σ .

The particles in \mathbf{P}_t also possess velocities. The time increment being conveniently assumed to be unity, if \mathbf{x} is a solution and \mathbf{v} is its velocity, the particle is updated simply as,

$$\mathbf{x} = \mathbf{x} + \mathbf{v}. \quad (7)$$

The particles in the population possess a memory of the best previous value in terms of fitness, called the personal best, \mathbf{p} . Each particle has its own personal best. Furthermore, during each iteration, the fittest particle from the entire population is identified and stored. This is the global best, \mathbf{g} . The personal and global bests are used to update the velocities of each particle as follows,

$$\mathbf{v} = \chi\mathbf{v} + c_1(\mathbf{p} - \mathbf{x}) + c_2(\mathbf{g} - \mathbf{x}). \quad (8)$$

In the above equation, the quantity χ , which is less than unity, is called the constriction coefficient. It is applied to prevent the particles from acquiring very large velocities. The quantities c_1 and c_2 are the cognitive and social constants. The second term in the above equation, containing the cognitive constant, is associated with the particle's own memory, while the last term that contains the social constant, determines how the particle interacts the rest of the population. Together, the last two terms allow the velocity to reorient itself towards the two reference solutions, \mathbf{p} and \mathbf{g} .

The global best \mathbf{g} need not be uniquely determined as the best particle in the entire population¹. As a matter of fact, much research has focused on PSO where each particle has its own predefined neighborhood, and identifies its own global best from it. Another option that has been explored is to break up the population into non-overlapping clusters in each iteration, and allow each particle to pick a global best from its own cluster.

Multi-Objective Particle Swarm Optimization

There are several multi-objective variants of PSO (*cf.* Reyes-Sierra & Coello, 2005). However, Multi-objective PSO (MOPSO) is among the most popular ones (Coello, 2004). In addition to the population, MOPSO maintains a repository of all non-dominated solutions found. This repository is updated at the end of the iteration, where older solutions are replaced with newly discovered ones that dominate them. As no single best particle can be identified in a multi-objective framework, the global bests, \mathbf{g} , in Eqn. 8 above, are chosen from the repository of non-dominated solutions.

Again, as a multi-objective optimization technique, MOPSO incorporates additional features to impose diversity in its population. The specific method to do so, called the *adaptive hypergrid* method, is illustrated in Figure 3 (b). According to this method, the objective function space is partitioned by means of a hypergrid. Each partition Π_k , $k = 1, 2, \dots, K$, where K is the number of partitions, is defined in terms of lower and upper bounds l_{ik} and u_{ik} along each dimension $i = 1, 2, \dots, M$. The partitions are equally sized and non-overlapping. Additionally, the hypergrid is adaptive, as it resizes every iteration to be just enough to cover all the solutions in the repository.

A solution \mathbf{x} is contained in Π_k if and only if its image in Ω lies within those bounds. In other words, $\mathbf{x} \in \Pi_k$ iff for each $i = 1, 2, \dots, M$, $l_{ik} < f_i(\mathbf{x}) < u_{ik}$. We shall denote by $\Pi(\mathbf{x})$ the partition containing solution \mathbf{x} in the repository. The example shown in Figure 3 (b), has two objective functions ($M = 2$). Between \mathbf{x} and \mathbf{y} , the two solutions shown, the partition containing \mathbf{x} , $\Pi(\mathbf{x})$, contains two solutions, while the one containing \mathbf{y} , $\Pi(\mathbf{y})$, has three. According to the hypergrid method, \mathbf{x} is considered to be located in a less diverse region than \mathbf{y} , and will be preferred over the latter.

For each particle, the reference, \mathbf{g} , for it to move towards is determined in two stages. At first, a roulette wheel selection is carried out between the partitions, where the probability $p(\Pi_k)$ of selecting a partition Π_k is inversely proportional to the number of solutions inside it, which we denote as $|\Pi_k|$, as,

$$p(\Pi_k) = \frac{|\Pi_k|}{\sum_{l=1}^K |\Pi_l|}. \quad (9)$$

A particle is then chosen randomly from the selected, with equal probability, as the global best \mathbf{g} .

ARTIFICIAL IMMUNE SYSTEMS

Artificial immune systems have provided the backdrop for yet another class of nature inspired methods. In this scheme, solutions draw rough parallels with antibodies in the vertebrate immune system, and are subjected to similar processes as real antibodies, which are, *cloning* and *affinity maturation*. In cloning, the antibodies in the population are sorted, and best few antibodies selected. For each selected antibody, a number of clones (*i.e.* identical copies), is obtained. The number of clones may either be fixed, or inversely proportional to the antibodies' rank in the sorted list. In other words, if $r(\mathbf{x})$ is the rank of antibody \mathbf{x} , the number of clones is given by,

$$N_c(\mathbf{x}) = \left\lceil \frac{\beta P}{r(\mathbf{x})} \right\rceil, \quad (10)$$

where β is an algorithm constant, and $\lceil \cdot \rceil$ is the usual ceiling operator. This process is called cloning.

The clones are then subject to affinity maturation, where a random perturbation is imparted to each. This perturbation may be done as in Eqn. 3. However, instead of a fixed variance σ , the amount of perturbation may be inversely proportional to the fitness of the cloned antibody. This case makes it possible to subject the better antibodies to lesser perturbation than worse ones. As a rule of thumb, the amount of perturbation is usually higher than that in evolutionary algorithms; hence the random perturbation in artificial immune systems is referred to as *hypermutation*.

Multi-Objective Artificial Immune Systems

In the authors' view, no single method for multi-objective optimization based on the artificial immune system metaphor has emerged, that can be considered to be the most popular. There are a few attempts, noteworthy of them being MISA (Multi-objective Immune Systems Algorithm), (Coello Coello & Cortés, 2005). In this section, an overview of MISA is provided as an example of an algorithm belonging to this class.

MISA begins with an initial population of antibodies that are generated randomly. However, in order to ensure that the search space, Σ , is uniformly covered with random antibodies, it is divided into different regions, and antibodies are generated separately within each region. These are inserted into a primary memory, \mathbf{M}^* . A secondary memory \mathbf{M} of non-dominated antibodies is obtained from \mathbf{M}^* .

In each iteration, the antibodies in the secondary memory are cloned. Each antibody \mathbf{x} is assigned an initial value of the number of clones, N_c . The number of clones for each antibody in \mathbf{M} is determined by invoking diversity. The adaptive hypergrid method is used for this purpose. The partition, $\Pi(\mathbf{x})$, containing each antibody, \mathbf{x} , is considered. If it contains more than the average number of antibodies in each

partition, the number of clones to be produced by \mathbf{x} is halved to $N_c(\mathbf{x}) = \frac{1}{2} N_c$. On the other hand, if the antibody \mathbf{x} occupies a partition that has below average the number of antibodies, the total number of clones it produces is doubled, *i.e.* $N_c(\mathbf{x}) = 2N_c$. In this manner, the algorithm attempts to fill the sparser partitions with more number of clones, while reducing the clones in partitions that are densely populated. It should be noted that changing the number of clones to be produced in this manner does not affect the total number of clones that are produced which are placed in \mathbf{M}^* , whose size remains a fixed fraction of \mathbf{M} . The cloned antibodies in \mathbf{M}^* are subject to hypermutation. In order to ensure that the antibodies in \mathbf{M}^* are different than \mathbf{M} , each clone is subject to some hypermutation. The secondary memory \mathbf{M} for the next iteration is obtained by extracting the non-dominated set of $\mathbf{M} \cup \mathbf{M}^*$. In other words,

$$\mathbf{M} = \Phi(\mathbf{M} \cup \mathbf{M}^*). \quad (11)$$

When the secondary memory is full, exceeding its maximum capacity, the hypergrid method is again invoked to remove those that are in denser partitions.

HYBRID APPROACHES

As nature inspired algorithms maintain a population of solutions in each iteration, instead of only a single one, they are capable of searching multiple regions of the search space simultaneously. This property, combined with their stochastic nature, allows these algorithms to carry out an effective exploratory search. To better improve their performance, they can be combined with a local search technique, which can exploit local features around each solution, and direct the search process towards more promising regions within its neighborhood. Much recent research has shown that the proper balance of exploration and exploitation is necessary for an efficient search to be carried out. Thus, several hybrid approaches that combine a nature inspired algorithm with a local search have been carried out. Hybridization schemes for multi-objective optimization have recently made their appearance.

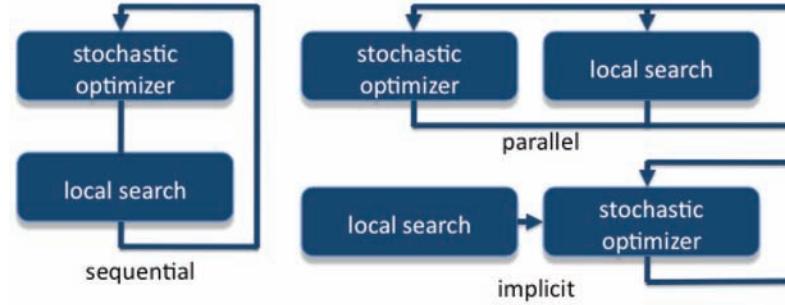
Hybridization Schemes

Hybridization can be divided into three basic approaches (Das, 2008): (*i*) *sequential*, (*ii*) *parallel*, and (*iii*) *implicit* (see Figure 4). In sequential hybridization, in each generation, the entire population undergoes the usual operations of the nature inspired stochastic algorithm in the usual manner. Following this, the solutions within the population are improved further using local search (Guo & Shouyi, 2003). Some approaches prefer to subject their entire populations to the local search, while others allow only the fittest one or more solutions to be improved further through this process.

In a parallel hybridization scheme, the population is split into two groups, which may or may not be overlapping. While the particles in one group follow the usual operations, the solutions in the other undergo local search. At the end of each generation, the two groups are merged together. Dividing the population is carried out randomly in some algorithms, while others prefer to place only the few best ones for further improvement.

In implicit hybridization, the local search is incorporated within one or more operator of the nature inspired algorithm. As an example, Das *et al.* include the direction provided by the local search as an

Figure 4. Three basic schemes for hybridizing stochastic algorithm with local search.



additional term in their velocity update equation (eqn. 8) in PSO (Das, Koduru & Welch, 2007). A crossover method that is based on the Nelder-Mead algorithm, described later in this section, has been proposed (Bersini, 2002).

Gradient Descent

Local search can either require the derivative of the objective function(s) to sense direction, or be derivative-free. Approaches that make use of the derivative of the objective function are gradient descent approaches. Here, the search is directed towards the negative gradient. In other words, the direction suggested, \mathbf{s} , is given by,

$$\mathbf{s} = -\nabla f(\mathbf{x}) = -\left[\frac{\partial f(\mathbf{x})}{\partial x_i} \right] \quad (12)$$

Although several multi-objective algorithms incorporate gradient descent, a general scheme to do so is lacking. Practically all hybrid methods apply local search separately along the objectives, f_1, f_2, \dots, f_M .

Nelder-Mead Simplex

Derivative free approaches for local search also exist. These methods are more versatile as they do not require the objective function(s) to be differentiable. A popular derivative free method for local search is the Nelder-Mead simplex. This method makes use of a set of $N+1$ solutions that are closely spaced, $S = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N+1}\}$, called the simplex. The worst solution, \mathbf{x}_w , in the simplex, is then identified, so that $f(\mathbf{x}_w) < f(\mathbf{x}_i)$ for all $\mathbf{x}_i \in S$. The centroid of the remaining solutions is determined as,

$$c = \frac{\sum_{i=1}^{N+1} \mathbf{x}_i - \mathbf{x}_w}{N} \quad (13)$$

In the next iteration, the new simplex is obtained by moving \mathbf{x}_w along the direction $c - \mathbf{x}_w$ to obtain a new solution,

$$\mathbf{x}_n = \mathbf{x}_w - \xi(\mathbf{c} - \mathbf{x}_w) \quad (14)$$

The new solution then replaces the worst one in the simplex for the next iteration. Thus,

$$S = S \cup \{\mathbf{x}_n\} - \{\mathbf{x}_w\}. \quad (15)$$

The quantity ξ is normally kept at unity. However, the simplex can be contracted and expanded by letting $\xi < 1$ and $\xi > 1$ respectively. Further details about the Nelder-Mead simplex algorithm and how it is incorporated within stochastic optimization methods can be found in (Das, 2008).

Fuzzy Dominance

The local search methods described above are inherently single objective methods. Fuzzy dominance is a method that can be applied to a set of solutions for any multi-objective optimization problem that assigns single values to each solution. The fuzzy dominances of the solutions can be treated as a measure of its overall fitness. The non-dominated solutions in a population are assigned zero fuzzy dominance. The further away from the non-dominated front that any given solution is, the closer will its fuzzy dominance be to unity. Thus using fuzzy dominance, it is possible to treat a multi-objective optimization problem, as if it only had a single objective. This concept has been used in the recently proposed algorithm, Fuzzy Simplex Genetic Algorithm (FSGA), which is a multi-objective optimization method that combines genetic algorithms as well as PSO with Nelder-Mead local search (Koduru, Das & Welch, 2007; Koduru *et al.*, 2008). We explain how fuzzy dominance can be computed.

Fuzzy dominance uses a *membership* function, $\mu(\cdot) : \mathbb{R} \rightarrow [0, 1]$. There are two requirements for this function: it must be monotonically increasing and it also must be zero for negative arguments. Given any two solutions \mathbf{x} and \mathbf{y} , within a population \mathbf{P} , in a problem involving M objectives, we can use this membership function to compare \mathbf{x} and \mathbf{y} , along any objective f_i . Specifically, if \mathbf{x} is better than \mathbf{y} along this objective, then the difference their evaluations for the objective, $f_i(\mathbf{x}) - f_i(\mathbf{y}) < 0$; hence $\mu(f_i(\mathbf{x}) - f_i(\mathbf{y})) = 0$ also. On the other hand, when \mathbf{x} is worse than \mathbf{y} , i.e. $f_i(\mathbf{x}) - f_i(\mathbf{y}) > 0$, then $\mu(f_i(\mathbf{x}) - f_i(\mathbf{y})) \geq 0$.

Note also that unless $\mathbf{y} \prec \mathbf{x}$, $\mu(f_i(\mathbf{x}) - f_i(\mathbf{y})) = 0$ for some objective. The pairwise fuzzy dominance $\mu(\mathbf{y} \prec \mathbf{x})$ between them is defined as,

$$\mu(\mathbf{y} \prec \mathbf{x}) = \mu(f_1(\mathbf{x}) - f_1(\mathbf{y})) * \mu(f_2(\mathbf{x}) - f_2(\mathbf{y})) * \dots * \mu(f_M(\mathbf{x}) - f_M(\mathbf{y})), \quad (16)$$

where the ‘*’ operator is a fuzzy *t*-norm, such as the minimum operator (*e.g.* $a * b = \min(a, b)$). Clearly, the pairwise fuzzy dominance, $\mu(\mathbf{y} \prec \mathbf{x})$, is nonzero only if $\mathbf{y} \prec \mathbf{x}$. It can also be seen that the pairwise fuzzy dominance provides a quantitative measure of degree of this dominance relationship: if \mathbf{y} marginally dominates \mathbf{x} , then $\mu(\mathbf{y} \prec \mathbf{x})$ is close to zero; on the other hand, if \mathbf{y} dominates \mathbf{x} by a large degree of severity, then $\mu(\mathbf{y} \prec \mathbf{x})$ is close to unity.

Using the pairwise fuzzy dominances between we can define the population fuzzy dominance of each solution in \mathbf{P} . Consider any such solution \mathbf{x} . If the pairwise fuzzy dominance $\mu(\mathbf{y} \prec \mathbf{x}) > 0$ with respect to any solution \mathbf{y} in \mathbf{P} , then $\mathbf{x} \notin \Phi(\mathbf{P})$ as $\mathbf{y} \prec \mathbf{x}$. Conversely, if $\mathbf{x} \in \Phi(\mathbf{P})$, then $\mu(\mathbf{y} \prec \mathbf{x})$ is necessarily

zero for each \mathbf{y} in \mathbf{P} . Let us define the population fuzzy dominance as,

$$\mu(\mathbf{P} \prec \mathbf{x}) = \mu(\mathbf{y}_1 \prec \mathbf{x}) \oplus \mu(\mathbf{y}_2 \prec \mathbf{x}) \oplus \dots \mu(\mathbf{y}_N \prec \mathbf{x}), \quad (17)$$

where each $\mathbf{y}_i \in \mathbf{P}$, and $\mathbf{y}_i \neq \mathbf{x}$, and the ‘ \oplus ’ is a t-conorm, such as the maximum operator, (e.g. $a \oplus b = \max(a, b)$). It can be seen that if $\mathbf{x} \in \Phi(\mathbf{P})$, then $\mu(\mathbf{P} \prec \mathbf{x}) = 0$, and moreover, when \mathbf{x} is dominated more severely by the others, the value of $\mu(\mathbf{P} \prec \mathbf{x})$ is close to unity.

Fuzzy dominance has been used to apply the Nelder-Mead algorithm within a genetic algorithm. The hybrid algorithm converges faster towards the Pareto front for several benchmark problems in multi-objective optimization, and uses a parallel hybridization scheme (Koduru *et al.*, 2008). Fuzzy dominance has also been successfully hybridized with a multi-objective version of PSO (Koduru *et al.* 2007), where it outperforms MOPSO and NSGA-II on several benchmarks.

CONCLUSION

We have provided an overview of the new and expanding field of multi-objective optimization, outlining some of the most significant approaches. We chose to describe NSGA-II and SPEA-2 as they are the most popular algorithms today. We also discuss the recent algorithm, ParEGO, which is very promising for some specialized applications as well as the even more recent FSGA, currently under development, which fills the need for hybrid multi-objective algorithms. Finally, we also have outlined MOPSO, which is based on a new evolutionary paradigm, PSO.

REFERENCES

- Bersini, H. (2002). The immune and chemical crossovers. *IEEE Transactions on Evolutionary Computation*, 6(3), 306–313. doi:10.1109/TEVC.2002.1011543
- Clerc, M. (2005). *Particle swarm optimization*. Washington, DC: ISTE Press.
- Coello, C. A. C. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3), 256–279. doi:10.1109/TEVC.2004.826067
- Coello, C. A. C., & Cortés, N. C. (2005). Solving multiobjective optimization problems using an artificial immune system. *Genetic Programming and Evolvable Machines*, 6(2), 163–190. doi:10.1007/s10710-005-6164-x
- Das, S. (2008). Nelder-Mead evolutionary hybrid algorithms. In J. Dopico, J. de la Calle, & A. Sierra (Eds.), *Encyclopedia of artificial intelligence* (pp. 1191-1196). Hershey, PA: Information Science Reference.
- Das, S., Koduru, P., Welch, S. M., Gui, M., Cochran, M., Wareing, A., & Babin, B. (2006). Adding local search to particle swarm optimization. In *Proceedings of the World Congress on Computational Intelligence*, Vancouver, BC, Canada (pp. 428-433).

- Das, S., & Panigrahi, P. K. (2008). Evolutionary algorithms for multi-objective optimization. In J. Dopico, J. de la Calle, & A. Sierra (Eds.), *Encyclopedia of artificial intelligence* (pp. 1145-1151). Hershey, PA: Information Scienc Reference.
- de Castro, L. N., & Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, 6(3), 239–251. doi:10.1109/TEVC.2002.1011539
- Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley.
- Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2), 182–197. doi:10.1109/4235.996017
- Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley.
- Guo, G., & Shouyi, Y. (2003). Evolutionary parallel local search for function optimization. *IEEE Transactions on Systems . Man and Cybernetics Part-B*, 7(1), 243–258.
- Knowles, J. (2006). ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1), 50–66. doi:10.1109/TEVC.2005.851274
- Knowles, J. D., & Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation*, 8, 149–172. doi:10.1162/106365600568167
- Koduru, P., Das, S., & Welch, S. M. (2007). Multi-objective and hybrid PSO using ϵ -fuzzy dominance. In D. Thierens, et al. (Eds.), *Proceedings of the ACM Genetic and Evolutionary Computing Conference*, London, UK (pp. 853-860).
- Koduru, P., Das, S., Welch, S. M., & Roe, J. (2004). Fuzzy dominance based multi-objective GA-Simplex hybrid algorithms applied to gene network models. In K. Deb, et al. (Eds.), *Proceedings of the Genetic and Evolutionary Computing Conference*, Seattle, Washington (LNCS 3102, pp. 356-367). Berlin, Germany: Springer-Verlag.
- Koduru, P., Das, S., Welch, S. M., Roe, J., & Lopez-Dee, Z. P. (2005). A co-evolutionary hybrid algorithm for multi-objective optimization of gene regulatory network models. In *Proceedings of the Genetic and Evolutionary Computing Conference*, Washington, DC (pp. 393-399).
- Koduru, P., Dong, Z., Das, S., Welch, S. M., & Roe, J. (2008). Multi-objective evolutionary-simplex hybrid approach for the optimization of differential equation models of gene networks. *IEEE Transactions on Evolutionary Computation*, 12(5), 572–590. doi:10.1109/TEVC.2008.917202
- Koduru, P., Welch, S. M., & Das, S. (2007). A particle swarm optimization approach for estimating confidence regions. In D. Thierens, et al. (Eds.), *Proceedings of the Genetic and Evolutionary Computing Conference*, London, UK (pp. 70-77).
- Mitchell, M. (1998). *An introduction to genetic algorithms*. Cambridge, MA: MIT Press.

Renders, J. M., & Flasse, S. P. (1998). Hybrid methods using genetic algorithms for global optimization. *IEEE Transactions on Systems . Man and Cybernetics Part-B*, 28(2), 73–91.

Reyes-Sierra, M., & Coello, C. A. C. (2006). Multi-objective particle swarm optimizers: A survey of the state-of-the-art. *International Journal of Computational Intelligence Research*, 2(3), 287–308.

KEY TERMS AND DEFINITIONS

Elitism: A strategy in evolutionary algorithms where the best one or more solutions, called the elites, in each generation, are inserted into the next, without undergoing any change. This strategy usually speeds up the convergence of the algorithm. In a multi-objective framework, any non-dominated solution can be considered to be an elite.

Exploration: A strategy that samples the fitness landscape extensively to obtain good regions.

Exploitation: A greedy strategy that seeks to improve one or more solutions to an optimization problem to take it to their closest optima.

Fitness: A measure that is used to determine the goodness of a solution for an optimization problem. In minimization problems, it is inversely related to the objective function.

Fitness Landscape: A representation of the search space of an optimization problem that brings out the differences in the fitness of the solutions, such that those with good fitness are “higher”. Optimal solutions are the minima of the fitness landscape.

Generation: A term used in nature inspired algorithms that roughly corresponds to each iteration of the outermost loop, where the existing population is replaced with a new one.

Local Search: A search algorithm to carry out exploitation. Pure local search algorithms are prone to getting trapped in local optima, without converging to good solutions.

Objective Function: The function that is to be optimized. In multi-objective optimization, several such functions need to be optimized simultaneously.

Population Based Algorithm: An algorithm that maintains an entire set of candidate solutions, each solution corresponding to a unique point in the search space of the problem.

Stochastic Algorithm: A non-deterministic algorithm, which relies on probabilistic operations. Natural algorithms that borrow from natural metaphors are almost always stochastic algorithms.

ENDNOTE

¹ Although this solution may be better described as *local best*, or *cluster best*, we will continue to refer to it as *global best* throughout this paper.

Chapter 5

Artificial Immune Systems for Anomaly Detection

Eduard Plett

Kansas State University at Salina, USA

Sanjoy Das

Kansas State University, USA

Dapeng Li

Kansas State University, USA

Bijaya K. Panigrahi

Indian Institute of Technology, India

ABSTRACT

This chapter introduces anomaly detection algorithms analogous to methods employed by the vertebrate immune system, with an emphasis on engineering applications. The basic negative selection approach, as well as its major extensions, is introduced. The chapter next proposes a novel scheme to classify all algorithmic extensions of negative selection into three basic classes: self-organization, evolution, and proliferation. In order to illustrate the effectiveness of negative selection based algorithms, one recent algorithm, the proliferating V-detectors method, is taken up for further study. It is applied to a real world anomaly detection problem in engineering, that of automatic testing of bearing machines. As anomaly detection can be considered as a binary classification problem, in order to further show the usefulness of negative selection, this algorithm is then modified to address a four-category problem, namely the classification of power signals based on the type of disturbance.

INTRODUCTION

The vertebrate immune system possesses the ability to recognize, adapt to, and eventually eliminate invasive foreign bodies with remarkable precision. Because of this unique capability, the immune system provides the basis for a number of bio-inspired problem solving approaches in engineering (Castro & Timmis, 2003). These approaches are collectively called *Artificial Immune Systems* (AIS).

DOI: 10.4018/978-1-60566-766-9.ch005

The first stage of the immune system's response to these foreign bodies is their detection. The task of recognizing foreign bodies is done by means of a class of cells called lymphocytes, which are present in the bloodstream. Two kinds of lymphocytes are present, the *B-cells* and the *T-cells*. The B-cells are produced by the bone marrow while the T-cells are generated by a structure called the thymus. The latter cells produce molecules called *antibodies*, which have the ability to bind themselves to specific molecules called *pathogens* that are found in the invasive foreign bodies. Depending on their structure, different antibodies will bind to different types of pathogens, and this ability is called the *affinity* of the antibodies. Further, antibodies must not bind to the molecules produced by their own organism. The ability to distinguish between own cells (called *Self*) and pathogens (called *Nonself*) is termed *Self-Nonself discrimination*. Self-Nonself discrimination is a key feature of the antibodies, and the principles of Self-Nonself discrimination have been successfully applied to many AIS based anomaly detection applications in engineering and computer science (*cf.* Aickelin *et al.* 2004).

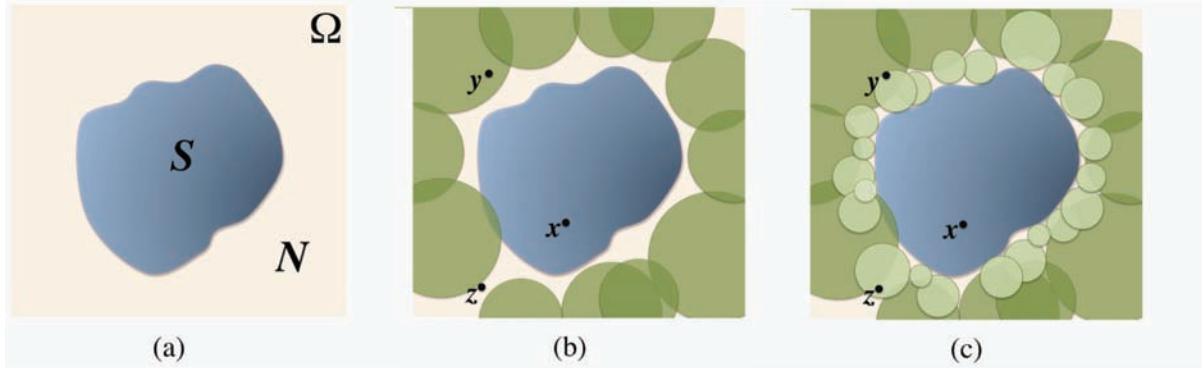
Under normal circumstances, Self-Nonself discrimination could work by using *positive characterization*, i.e. train the antibodies to recognize known samples of foreign cells. For example, a typical application of positive characterization is a computer anti-virus program. The virus definitions have to be periodically updated to enable the system to identify new threats. However, it is impossible to predict all possible contaminations by foreign cells, and such a system would be unable to react to threats which has not encountered before. The vertebrate immune system is, therefore, based on negative characterization or *negative selection* instead (Aickelin *et al.* 2004; Luh & Chen, 2005). This is accomplished by continuously creating a large variety of antibodies. These antibodies are then presented to the body's own cells. If an antibody is found to bind to any of the latter cells, it is simply eliminated from the bloodstream. This is done so that the immune system does not develop an adverse autoimmune reaction. Otherwise, the antibody is released in the bloodstream. The detection process is then straightforward: if an antibody binds to any cell, it is assumed to be foreign and is then destroyed by the immune system.

Artificial immune system algorithms based on negative selection are the mainstay of anomaly detection methods. In a manner reminiscent of their biological counterpart, these algorithms generate a repertoire of *detectors*. These detectors are generated in substantial numbers with the expectation of covering the entire Nonself region. Subsequently, any input that is detected by any detector is classified as an anomalous input.

Negative selection algorithms have been successfully applied to many anomaly detection problems. Probably the most intuitive applications are in enhanced computer security (Dasgupta & Gonzales, 2002; Harmer *et al.*, 2002; Nia *et al.*, 2003). Other applications use negative selection to detect faults in squirrel cage induction motors (Branco *et al.*, 2003), refrigeration systems (Taylor & Corne, 2003), aircraft systems (Dasgupta *et al.*, 2004), and power systems (Gui *et al.*, 2007). They also have been used to detect anomalies in time series data (Nunn & White, 2005) and to recognize patterns, for example the Indian Telugu characters (Ji & Dasgupta, 2004; Ji & Dasgupta, 2006). More recently, a method that uses negative selection in conjunction with an optimization algorithm has been applied to classify faults in rotor rigs from vibration data (Strackeljan & Leiviskä, 2008).

This chapter begins with a description of the basic idea behind negative selection. Next, it focuses on a subclass of detectors, called *V-detectors*, or variable-sized detectors, which is a popular choice in many recent engineering applications. Several recent methods derived from basic negative selection have been outlined, which are divided into three broad categories: self-organizing detectors, evolving detectors and proliferating detectors. In order to demonstrate the effectiveness of negative selection, this chapter describes in detail the proliferation mechanism and proposes extending the detector proliferation

Figure 1. (a) Self (S) and Nonself (N) regions in Ω . (b) Initial detector repertoire after negative selection. (c) Further detectors after proliferation.



method beyond anomaly detection to also perform multi-category classification.

As a case study, the V-detector and proliferating V-detectors algorithms are applied to a real-world engineering problem of detecting anomalous data for an automated bearing testing machine. The goal of the application is to determine when a bearing is near the end of its useful life, based on several observed parameters, some of which may migrate outside of their normal ranges of operation towards the end of the product's normal lifetime.

As a second case study, this chapter also demonstrates the use of the proliferating V-detectors algorithm to solve a power quality disturbance classification problem. Four normalized, statistical features for power quality disturbances are extracted from raw data using the S-transform technique. The proposed classification method, based on the proliferating V-detectors algorithm, is then applied.

Simulation results for both case studies verify the effectiveness of the proposed method, which is based on negative selection.

NEGATIVE SELECTION

In order to detect anomalies in any equipment or process, its operation has to be monitored continually, and samples of observed variables collected at regular intervals of time. This data is then applied as inputs to the anomaly detection system for detection of abnormal behavior. The set of all possible inputs is called the input space, which is divided into Self and Nonself regions. Normal and anomalous data correspond to points in the Self and Nonself regions, which we shall denote as S and N respectively, such that $S \cup N = \Omega$, the input space, and $S \cap N = \emptyset$. The relationship between S , N and Ω is depicted in Figure 1 (a).

During the negative selection process, a large number of random detectors are generated, in a mostly random manner. Self is then presented to each detector to see if the detector erroneously detects it. If it does, the detector is discarded; otherwise it is inserted into the detector repertoire. This process is carried out until enough valid detectors are found to ensure full coverage of the Nonself region. Such a repertoire of detectors is shown in Figure 1 (b), with each detector being represented as a shaded circle in N . The same figure also shows a sample point, x , inside S , and two others, y and z , outside it. Although y can

be detected by one of the detectors, there may be a few such as z , which cannot be detected. Figure 1 (c) shows the additional step of proliferation, which can be applied to provide additional coverage. We postpone a discussion of the proliferation mechanism until later.

After the detector repertoire is complete, the anomaly detection system can be put to use by receiving external inputs and presenting the inputs to the individual detectors. If the input lies in the vicinity of an existing detector, the detector detects that input. In this case, the input is classified as an *anomaly*. Conversely, an input that is not detected by any of the detectors in the repertoire is regarded as *normal*. In order to ensure that the anomaly detection system can catch any anomaly, the entire Nonself region has to be covered with detectors. For this purpose, a number of schemes have been proposed to rapidly fill up that region in a computationally efficient manner, which are addressed in the next section.

Representation Schemes

In order to represent detector parameters as well as inputs: they are formatted as strings. Strings can be composed of numbers, alphanumeric symbols, Boolean values, etc. In many applications, such as computer intrusion detection, simple binary strings are used to represent detectors and inputs. An input is considered anomalous if certain bits or a group of bits in the input string match the corresponding detector bits. In other applications, for example for detecting misbehaving nodes in a mobile *ad hoc* network (Sarafijanovic & Le Boudec, 2005), more complicated schemes are used to organize combinations of events in “genes”, then thresholds are used to encode them in binary strings. Yet in other applications, for example to detect event sequences that lead to fatal errors in ATM machines (de Lemos *et al.*, 2007), the states are represented as integers. An error sequence is detected if a certain sequence of states in an input string matches the corresponding sequence in a detector string.

Real-valued representations are a more specific format, where the string is entirely composed of real or integer values only. Real valued representations are useful mainly in engineering applications where input data is typically real-valued also, such as temperature of an equipment, or frequency of rotation of rotating machinery. Each input can therefore be construed as a multi-dimensional vector or point. Further details of real-valued detectors, which are the focus of the present chapter, are addressed below.

Real-Valued Detectors

Hyperspheres are a particularly popular choice to represent real-valued detectors (Nio *et al.*, 2003; Ji & Dasgupta, 2004; Das *et al.* 2008). In this scheme each detector has two basic parameters, a center, *i.e.* its location in Nonself (N), as well as a radius. For the i^{th} detector, we shall denote them as $\boldsymbol{\mu}_i$ and r_i respectively. If $W \subset \Re^D$, where D is the dimensionality of the input space, the center is a D dimensional vector, $\boldsymbol{\mu}_i \equiv [\mu_{i1}, \mu_{i2}, \dots, \mu_{iD}]$.

Any input to the anomaly detection system, \mathbf{x} , also D dimensional, is said to be detected by the i^{th} detector if the following condition is true:

$$\|\mathbf{x} - \boldsymbol{\mu}_i\| \leq r_i \quad (1)$$

In the above equation, the operator $\|\cdot\|$ is the Euclidean norm. Thus, for an input to be detected, it must be located at a distance not more than the radius r_i .

In many practical situations, a complete description of Self (S) is difficult to obtain. In such cases, instead of S , a finite set of normal sample data s_j ($j = 1, 2, \dots$) in S , obtained experimentally, may be presented to the anomaly detection system. In order to properly define S from these samples, a parameter γ , called the Self radius, is used. In a manner similar to the detectors, the region enclosed by each hypersphere of radius γ , and centered around each s_j , is considered to be a part of S . The latter can then be defined as the union of such regions. In other words,

$$S = \left\{ \mathbf{x} \mid \exists j \text{ such that } \|\mathbf{x} - s_j\| \leq \gamma \right\} \quad (2)$$

For the negative selection algorithm to fully cover N , we would ideally like to have the Nonself region defined directly in terms of the detectors as:

$$N = \left\{ \mathbf{x} \mid \exists i \text{ such that } \|\mathbf{x} - \mu_i\| \leq r_i \right\} \quad (3)$$

Multi-shape detectors and variations of hyperspheres have been considered extensively (Balachandran *et al.*, 2007). For example, hyperspheres have been generalized to hyperellipses of the form $(\mathbf{x} - \mu_i)\Lambda(\mathbf{x} - \mu_i)^T \leq r_i$, where Λ is a correlation matrix, and other quantities have their usual meaning as described earlier. Another form of multi-shaped detector has been considered by replacing the condition shown in equation (1) with $\|\mathbf{x} - \mu_i\|_p \leq r_i$, where $\|\mathbf{x} - \mu_i\|_p$ is a p -norm. For the special cases when $p = \infty$, the regions are shaped as hypercubes. For other values within the range $(2, \infty)$, the shapes of the regions are intermediate between hyperspheres and hypercubes. The advantage of using such shapes as detectors is increased coverage. Shapes other than hyperspheres are often necessary to fill the Nonself region when the boundary between Self-Nonself regions is complex. An extreme situation where S is in a fractal shape is addressed by Das *et al.* (2008). However, in that study, instead of using multi-shaped detectors, the proliferating V-detectors method, which will be described later, was used. Although in typical engineering systems, S might not be so complex, the application of proliferating V-detectors for such cases provides other benefits also, including surer detection of anomalies, as has been shown in Das *et al.* (2008).

The V-Detectors Algorithm

Within the real-valued detection schemes, a common version of the negative selection algorithm is the V-detectors algorithm where the detectors are hyperspheres of variable radii (Ji & Dasgupta, 2004, Das *et al.* 2008). It proceeds in an iterative manner. Within each iteration i , a random point, $\mathbf{x} \in \mathbb{C}$, is generated by the algorithm. If this point satisfies the condition in equation (2), then it is within S and is discarded. On the other hand, if the same condition is not met, a new detector is created with its center $\mu_i = \mathbf{x}$. The radius r_i is made as large as possible, without overlapping S . This is achieved by letting r_i be given by:

$$r_i = \min \|\mathbf{x} - s_j\| - \gamma \quad . \quad (4)$$

The negative selection algorithm is terminated when a large number of detectors are generated, so that the condition in equation (3) is satisfied to the fullest possible extent. While this approach conforms

to vertebrate biology and can be applied to generate detectors, a number of algorithmic improvements have been proposed in the recent literature. These methods enable detectors to fill up the Nonself region with reasonable computational costs and are taken up next.

EXTENSIONS OF NEGATIVE SELECTION

In recent years, several schemes have been proposed to achieve better coverage than what is possible through simple negative selection mechanism alone. One of the contributions of this chapter is to propose a classification of all these methods into three basic approaches. The first approach is to seek a rearrangement of the detectors in the repertoire to maximize coverage with the existing detectors only. This is done through the process of self-organization of their centers. The second scheme aims to improve the negative selection algorithm by incorporating an evolutionary algorithm, which replaces random generation with a more optimal means to create detectors to be introduced into the repertoire. The third method aims to use existing detectors to create new ones, which as mentioned previously, attempts to better fill the hard-to-reach locations in the boundary between S and N .

Detector Self-Organization

The motivation behind this approach is to seek a rearrangement of the positions of existing detectors in the repertoire so as to maximize the coverage of the Nonself region without having to introduce newer ones. The basic idea is to spatially reorganize the detectors by pushing them away from one another, so that the overlapping region covered by neighboring detectors, as defined by the intersection of their hyperspheres, is minimized. This idea, well studied by Dasgupta & Gonzales (2003), restricts the detectors to hyperspheres of fixed radii, r . This restriction simplifies the algorithm by avoiding the explicit computation of the overlapping regions between hyperspheres, and allows self-organization to be based on Euclidean distances between the detector centers only. The center μ_i of each detector i is moved along a direction d_i as in the following equation:

$$\mu_i = \mu_i + \eta d_i \quad (5)$$

Here the quantity η is a quantity associated with the self-organization process. Starting with a relatively high value, this quantity is constantly lowered as the detector placement continuously improves with increasing iterations. For the j^{th} detector, the direction vector must be directed away from μ_j . Hence it is determined as a weighted sum of the vectors $\mu_i - \mu_j$ in the following manner:

$$d_i = \frac{\sum_{j \neq i} \zeta_{j,i} (\mu_i - \mu_j)}{\sum_{j \neq i} \zeta_{j,i}} \quad (6)$$

In the above equation, the quantity $\zeta_{j,i}$ is a weight associated with the j^{th} detector. The denominator is used for normalization. Since detectors that are too far away need not influence the movement of detector i , this weight decreases exponentially with increasing distance between the detectors under

consideration according to:

$$\varsigma_{j,i} = e^{-\alpha \|\mu_i - \mu_j\|^2} \quad (7)$$

In the above, the quantity α is simply an algorithmic constant. The self-organization algorithm proposed by Dasgupta & Gonzales (2003) begins with a repertoire of detectors whose centers μ_i in the input space are generated randomly. It extends the idea of rearranging detectors to include those that are inside S . Instead of removing them from the repertoire, as in pure negative selection, these detectors are pushed away from S using equation (6). However, the direction d_i is computed as follows:

$$d_i = \frac{\sum_j \zeta_{j,i} (\mu_i - s_j)}{\sum_j \zeta_{j,i}} \quad (8)$$

The weight $\zeta_{j,i}$ is now associated with the j^{th} Self sample. It is determined by first obtaining the k nearest neighbors of μ_i from the set of Self samples, s_j . Denoting the set of these neighbors as B_i , the weight is defined as:

$$\zeta_{j,i} = \begin{cases} 1 & s_j \in B_i \\ 0 & \text{otherwise.} \end{cases} \quad (9)$$

All the original detectors of the repertoire are not always retained. When a detector is so deeply buried inside S that pushing it towards the periphery as in equation (5) is computationally expensive, it should be removed. In order to identify such detectors, each detector is associated with a variable, l_i , called its life, which is initialized to zero. It is incremented each time the detector is repositioned using equation (5). When the life exceeds a certain predetermined threshold, the corresponding detector is deleted.

Evolving Detectors

Evolutionary algorithms have been quite successful in augmenting negative selection to generate a repertoire of detectors with wide coverage (Dasgupta & Gonzales, 2002; Balachandran *et al.*, 2007; Gao *et al.*, 2008). The overall strategy in these cases is to subject each detector to an evolutionary process to improve its coverage before adding it into the repertoire of detectors. Starting with an initially empty repertoire, this process is repeated until an adequate number of detectors have been added to it.

In order to add a new detector to the detector repertoire using an evolutionary algorithm, the first step is to generate an entire population of random candidate detectors. These are subject to the standard variational operators of crossover and mutation used in evolutionary algorithms to generate even more detectors. The fittest (best) detectors are then selected to comprise the new population. This process is repeated several times. Upon termination, the fittest detector of the population is selected for inclusion in the anomaly detection system's repertoire of detectors.

The fitness of each new detector in such evolutionary schemes is formulated to reflect the additional

coverage gained by introducing it to the existing repertoire of detectors. A typical form of this fitness has been suggested as follows (Dasgupta & Gonzales, 2002; Balachandran *et al.*, 2007):

$$f_i = v_i - \sum_j n_{i,j} \quad (10)$$

In the above equation, f_i is the fitness of the i^{th} new detector, v_i is its hypervolume, and $n_{i,j}$ is the hypervolume of the region formed by the intersection of that detector and the j^{th} detector from the existing repertoire.

Detector Proliferation

Detector proliferation is the third basic mechanism of accomplishing a wider coverage. Using the proliferation mechanism, detectors can be created in a computationally more efficient manner using ones already existing in the repertoire. In this method, at first, an initial repertoire of detectors is generated through the V-detectors algorithm. These detectors are then allowed to proliferate, i.e. create *offspring* detectors. An offspring detector k is created directly by taking an existing detector i , and determining the center \mathbf{x}_k as,

$$\mathbf{x}_k = \mathbf{x}_i + \eta r_i \hat{\mathbf{u}}. \quad (11)$$

Here, $\eta \in \mathbb{R}^+$ is a quantity called the *reachability* coefficient that determines how far away the offspring is from its parent, and $\hat{\mathbf{u}}$ is a unit vector (*i.e.* $\|\hat{\mathbf{u}}\| = 1$). When $\eta = 1$, the center of \mathbf{x}_k is on the surface of the parent detector. In order to enable the offspring to be further away from the parent, a value of $\eta > 1$ can be used. On the other hand, when $\eta < 1$, the offspring is closer to the parent. Das *et al.* (2008) have used a value of unity in power systems anomaly detection algorithms. The radii of the offspring detectors can be determined in accordance with equation (4).

Since detectors created through the V-detectors method border on the Self region S (see equation (4)), some of the offspring detectors end up even closer to the $S-N$ boundary. Others more effectively fill the ‘gaps’ between detectors. Since no offspring detectors are ever discarded in this method, a fewer number of iterations are needed to establish an equivalent level of coverage in comparison to the original V-detectors algorithm.

Detector proliferation can also be implemented in stages. In the first stage of proliferation, the detectors generated by the V-detectors method are used as parents. In each subsequent stage, the offspring of the previous stage become the parents of the current stage. Stage-wise proliferation is shown to produce more accurate anomaly detection (Das *et al.*, 2008).

Later in this chapter, we demonstrate the effectiveness of the V-detectors and the proliferating V-detectors algorithms for the automated detection of anomalies in a bearing testing machine. However, we first extend the proliferating V-detectors algorithm to classification applications.

Extending Detector Proliferation for Classification Applications

Anomaly detection can be seen as a 2-category classification problem, where the task is to classify any input into one of two possible classes: Self or Nonself. This section proposes the idea of applying a variant of stage-wise proliferation to accomplish multi-category classification. Any traditional K -category classification scheme is regarded as a $K+1$ category classification method here, with the inclusion of Nonself N as a new category, which, for convenience, we will number as the 0th category. In such classification schemes, the training data contains separate samples belonging to each category. Let us denote by $\mathbf{s}_j^{(k)}$ the j^{th} sample belonging to the k^{th} category. Using the V-detectors algorithm, samples from all K categories are used to create Nonself detectors, whose centers and radii are denoted as $\mu_i^{(0)}$ and $r_i^{(0)}$, $i = 1, 2, \dots$. Thus, we now redefine the Nonself region as:

$$N = \left\{ \mathbf{x} \mid \exists i \text{ such that } \|\mathbf{x} - \mu_i^{(0)}\| \leq r_i^{(0)} \right\}. \quad (12)$$

Note that the above equation resembles the earlier description of Nonself in equation (3), the difference being the introduction of superscripts to denote the category of the detectors.

Next, the algorithm goes through separate stages of proliferation, once for each category, $k = 1, 2, \dots, K$. During the k^{th} proliferation stage, samples from the k^{th} category are removed. The gap created by this removal is filled with offspring detectors. Only the original repertoire of detectors, which were produced by the V-detectors algorithm are used for proliferation. The center and radius of the j^{th} such detector created from the i^{th} parent are given by:

$$\mu_j^{(k)} = \mu_j^{(k)} + \eta r_i^{(0)} \hat{\mathbf{u}}, \quad (13)$$

and

$$r_j^{(k)} = \min \left\| \mu_j^{(k)} - \mathbf{s}_l^{(h)} \right\| - \gamma. \quad (14)$$

In the above equation, $\mathbf{s}_l^{(h)}$ is a Self sample from category h . Since the Self samples from all other categories other than category k are considered to compute the radius, $h \in \{1, 2, \dots, K\} - \{k\}$. The $\min(\cdot)$ operator is applied to all such hs and ls .

The region defining the k^{th} category in the input space W is thus:

$$S^{(k)} = \left\{ \mathbf{x} \mid \exists j, k \text{ such that } \|\mathbf{x} - \mu_j^{(k)}\| \leq r_j^{(k)} \right\}. \quad (15)$$

Later in this chapter, the effectiveness of this method is illustrated for the case of a 4-category power signal classification problem.

CASE STUDY 1: ANOMALY DETECTION IN A BEARING TESTING MACHINE

Problem Description

An automatic bearing testing machine performs accelerated testing of bearings for heavy-duty construction equipment. The machine simulates operation in the field by applying cyclical forces to a bearing while it is subject to rotation and contamination with dirt, sand and water. The load cycles are counted and are a measure of the life expectancy of a bearing. Initially, a baseline test is performed with a standard production bearing. New bearing designs are then tested under identical conditions. Only bearings that show significant improvement in life expectancy over the baseline bearings are then subjected to a much more expensive field test for final validation.

A critical part of the test procedure is to correctly determine when a bearing reaches the end of its useful life. Without taking the bearing apart, this can be done by relying on data, which are recorded at specific time intervals during the test. A total of 19 parameters are recorded, corresponding to 4 different bearings which are tested simultaneously. The three most important parameters are: the wear (play) of the bearing (recorded in inches), the force that it takes to rotate the bearing (recorded in pounds), and the temperature of the bearing (recorded in °F).

Cumulatively, these three parameters are a very good indicator of the status of the bearing. At the beginning of the test, the normal region of operation N for the different test variables is established. As the test progresses, the real-time measurements are compared to the baseline numbers and an automatic shutdown is implemented by the machine control system if any of the variables violates its limits.

The problem with this type of *traditional limit-checking* algorithm is that each variable is treated as independent from all the others, which it is not. The algorithm only checks if an individual variable violates its limit. It essentially surrounds the normal region with a rectangle (two dimensions), cube (three dimensions) or hypercube (more than three dimensions) and stops the test if any of the planes of the hypercube are violated.

An algorithm of this type is able to detect sudden changes (*e.g.* breakage), and bearings which are far beyond their useful life. However, treated as a single interdependent body in three dimensions, the readings leave the normal area of operations and migrate towards the Nonself areas inside the limit cube well before the variables violate their individual limits. A human operator can often easily detect this condition, and therefore in practice an operator - instead of the machine - usually makes the call to stop the test. The goal of this application is then to give the machine a similar type of intelligence, *i.e.* make it able to detect when the system leaves the normal region of operation. Negative selection based algorithms are perfectly suited for this type of application because they do not treat individual inputs as independent from each other, but as a complex “body”. They are therefore able to detect multidimensional anomalous inputs in the border regions where none of the individual limits in any particular direction are yet violated.

When choosing an appropriate algorithm for a real-time control system, two important constraints have to be considered. One constraint is the typically small controller memory, too small to store a large detector repertoire, especially when dealing with multidimensional, real-valued, floating-point data. The second constraint is the scan time of the controller, which might increase to unacceptable levels due to the significantly larger computation overhead required for negative selection algorithms with several hundred or even thousand detectors. Therefore, the detector repertoire has to be as small as possible, which eliminates many potential negative selection based methods. Therefore, only the traditional V-

detectors algorithm and the proliferating V-detectors algorithm were considered for this problem.

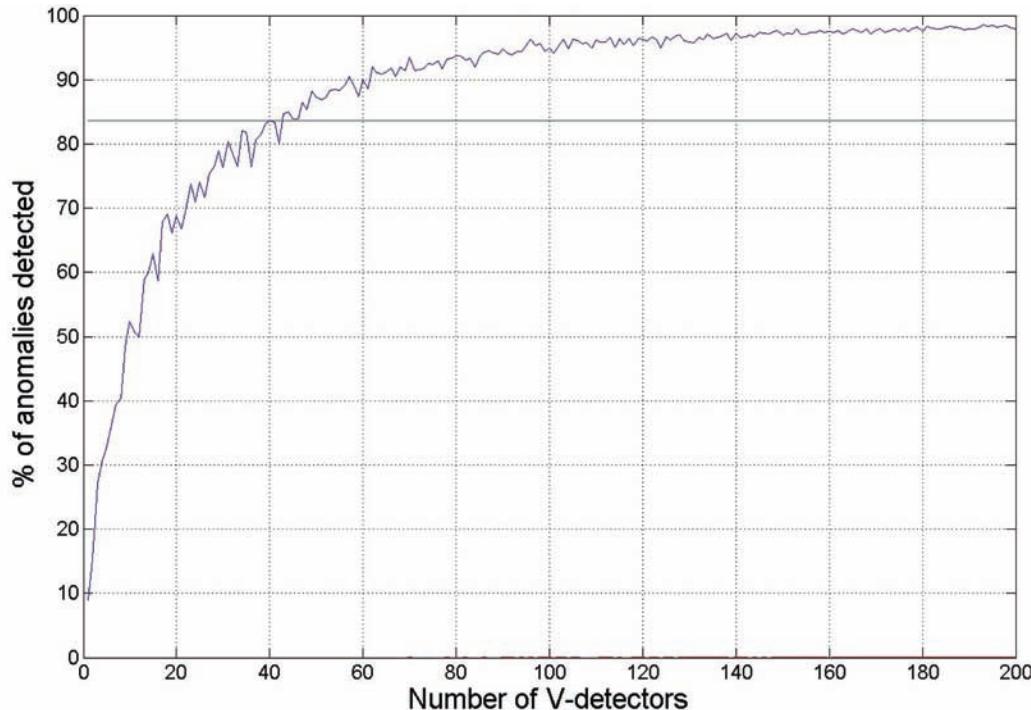
Based on the normal data samples, the detector repertoire and the associated radii are computed offline according to the V-detectors algorithm. Once a good detector repertoire is found, it is stored in controller memory in a look-up-table format. The controller, in addition to the usual operations of reading inputs, making decisions based on inputs, and updating outputs, also computes the Euclidian distance between the corresponding inputs and the detectors, and alerts the operator if the data readings leave the normal region of operation.

Results

To validate the detector repertoire, a data set of 1600 data points are imported into MatlabTM, where the first 200 data points represent the “break in” time of the bearing, when the measurements tend to wander back and forth. The next 1000 data points represent the normal or Self region of operation. The last 400 data points represent the end-of-life region where the measurements start to inch upwards, indicating a worn bearing. The mean and the standard deviation of each dimension in the normal region were found and all points where at least one dimension was three times larger or smaller than the standard deviation are considered an anomaly. The whole data set is then evaluated with the V-detector, the proliferating V-detector and the traditional limit-checking algorithms. All of these algorithms try to maximize the number of anomalies detected and minimize false detections while varying two parameters: the size of the Self radius and the number of the V-detectors.

Figure 2 shows the percentage of anomalies detected as the number of V-detectors varies from 1 to 200. As expected, the traditional limit-checking algorithm does not depend on the numbers of detectors,

Figure 2. Detected anomalies vs. number of V-detectors, basic algorithm



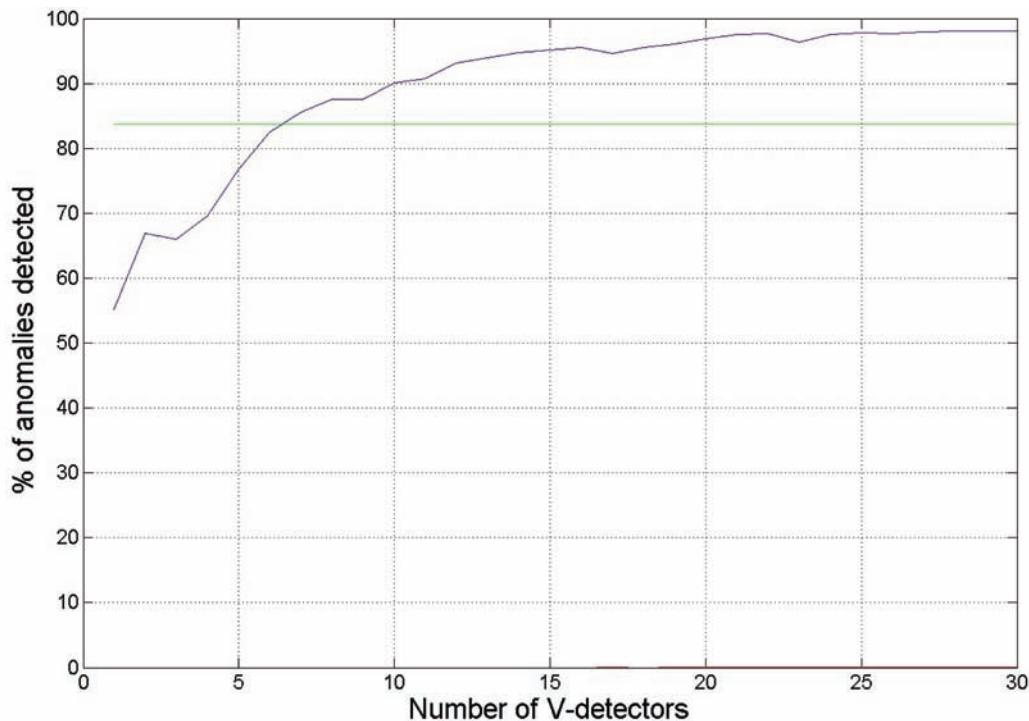
so it is a constant, depending solely on the gap between the Self area and the limit planes. To achieve the best possible detection rate, the gap was reduced to zero for this test, i.e. the limit planes are touching the outlying hyper-spheres of the Self area. Even with this very tight limit gap, the V-detectors algorithm outperforms the traditional limit checking algorithm starting at about 45 detectors. Figure 3 shows the percentage of anomalies detected as a function of the number of detectors, for the proliferating V-detectors algorithm, and follows a similar trend as before.

For the proliferating V-detectors algorithm, the number of parent detectors was varied between 1 and 30 and a single proliferation stage was used to also limit the size of the detector repertoire to around 200 ($30 + (30 \times 6) = 210$, to be exact). The performance was comparable to the randomly placed 200 detectors of the basic V-detectors algorithm, but it was achieved with only 30 iterations compared to 200 for the basic V-detectors algorithm.

Figure 4 shows the percentage of anomalies detected as the Self radius varies from 0.02 (three times the standard deviation) to 0.05. It is evident that as the Self radius, γ , increases, the number of anomalies detected decreases for the traditional limit-checking algorithm and V-detectors algorithms. This stems from the fact that, by definition, values larger than 3 times the standard deviation were considered anomalies, and by increasing the Self-radius some anomalous points ended up inside the Self area, and were therefore missed by the limit planes and by the detectors. However, the V-detectors algorithms still outperformed the traditional limit-checking algorithm for all Self-radii tested.

This application demonstrates that the V-detectors and proliferating V-detectors algorithms are a feasible and desirable approach for real-time data collection system evaluations. Both V-detectors algorithms always significantly outperform traditional limit-checking algorithms because V-detectors algorithms do

Figure 3. Detected anomalies vs. number of detectors, proliferating V-detectors



not treat individual inputs as independent from each other, but as a complex body, and so are capable of detecting anomalous points missed by the limit planes. The performance of a proliferating V-detectors algorithm was comparable to the performance of a basic V-detectors algorithm, provided they both were using a similar number of detectors. Essentially, the basic V-detectors algorithm used completely random detectors while the proliferating V-detectors algorithm uses strategically placed detectors. The advantage of the proliferating V-detectors algorithm was that enough detectors were obtained in a small fraction of the total number of iterations (30 instead of 200). In addition, we speculate that the advantage in detection ability of the proliferating V-detectors algorithm would be more distinct for when the Self region is continually evolving in time in the bearing test machine with changing environmental conditions.

CASE STUDY 2: POWER QUALITY SIGNAL CLASSIFICATION

Problem Description

Ideally, in an electric power delivery system, the power signal should be a pure sinusoid of a constant frequency. However, in practice, a number of factors result in the introduction of harmonic components. These harmonic components do not result in higher power being delivered to the user, but cause undesirable effects such as overheating, higher losses, and occasionally equipment damage. Although small harmonics that fall within acceptable limits are tolerated, larger ones that are introduced into the power signal due to the presence of disturbances are undesirable. Therefore, monitoring the quality of

Figure 4. Detected anomalies vs. Self-radius

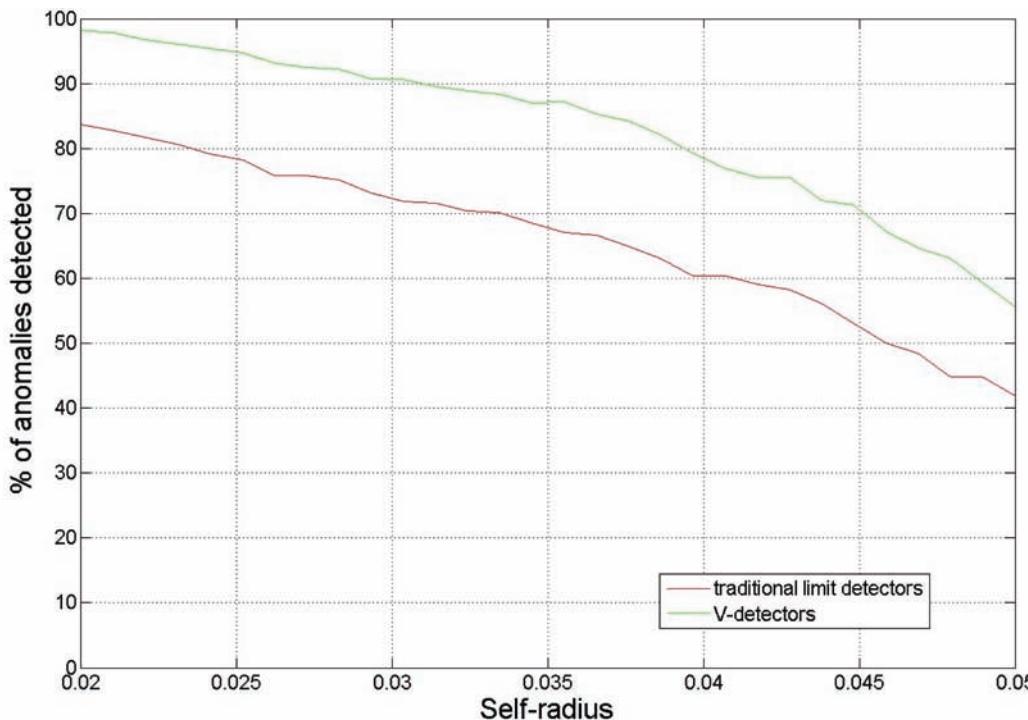
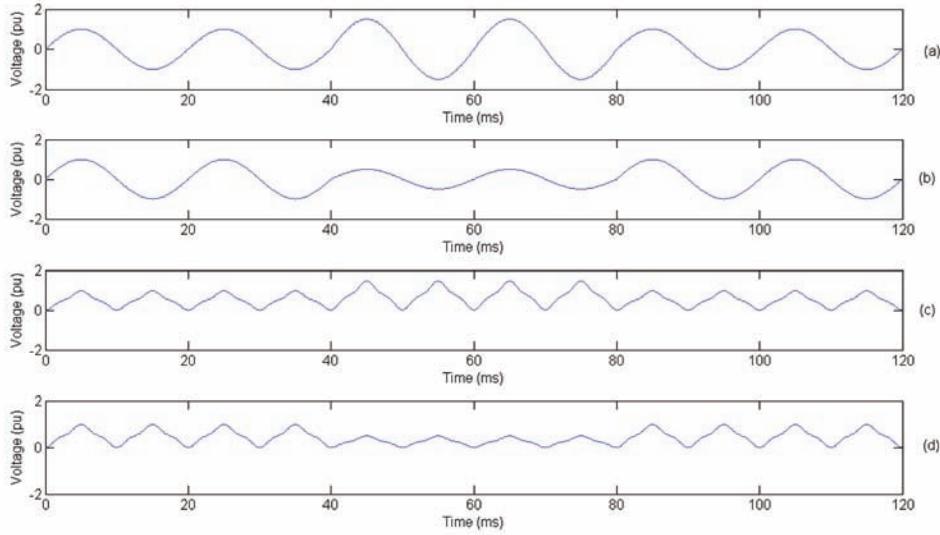


Figure 5. Raw input signals belonging to each disturbance.



power to detect large harmonics is of paramount importance in electric power systems. Further, being able to classify such disturbances from the power signal also helps in locating the underlying cause of the disturbance.

In order to further demonstrate the applicability of proliferating V-detectors for more general multi-category classification, an automated method for classifying disturbances from the power signals is described here. Before being presented to the V-detectors algorithm, the continuous time signal is subject to preprocessing using S-transforms to obtain input feature vectors.

Let $h(t)$ denote a continuous-time power quality signal. Samples of this signal at regular intervals of T are $h(kT)$, where $k = 0, 1, \dots, N-1$. The discrete Fourier transform of $h(kT)$ is:

$$H\left(\frac{n}{NT}\right) = \frac{1}{N} \sum_{k=0}^{N-1} h(kT) e^{-(2\pi nk/N)}, \quad (16)$$

where $n = 0, 1, \dots, N-1$. The inverse discrete Fourier transform is obtained by:

$$H(kT) = \sum_{n=0}^{N-1} H\left(\frac{n}{NT}\right) e^{2\pi nk/N}. \quad (17)$$

The S-transform consists of the projections of the vector defined by the time series $h(kT)$ onto a spanning set of vectors. Since the spanning vectors are not orthogonal, the elements of the S-transform are not independent. Each basis vector is divided into N localized vectors by an element-by-element product with N shifted Gaussian windows. Therefore, the S-transform of a discrete time series $H(kT)$ is given by:

$$S\left(jT, \frac{n}{NT}\right) = \sum_{m=0}^{N-1} H\left(\frac{m+n}{NT}\right) G(m, n) e^{\frac{2\pi m j}{N}} \quad (18)$$

where the Gaussian function $G(m, n) = e^{\frac{-2\pi m^2}{n^2}}$, and $j, m, n = 0, 1, \dots, N-1$.

From the matrix $S\left(jT, \frac{n}{NT}\right)$, a set of four features is computed. These are: (i) the standard deviation of the phase contour, (ii) the standard deviation of the data set comprised of the elements corresponding to maximum magnitudes of each column of the S-matrix, (iii) the standard deviation of the data set values corresponding to the maximum values of each row of the S-matrix, and (iv) the energy of the data set comprised of the elements corresponding to the maximum magnitudes of each column of the S-matrix. For further details of this data preprocessing, the reader is referred to the work of Dash *et al.* which also uses an S-transform approach (Dash *et al.* 2003).

Data pertaining to four different power quality disturbance signals are considered here. They are (i) voltage swell (Sw), (ii) voltage sag (Sg), (iii) voltage swell with harmonics (SwH), and (iv) voltage sag with harmonics (SgH). The sampling frequency is 3.2 kHz (64 samples per cycle). The number of samples of signals in each class are 50, 85, 50, and 51, respectively. Figure 5 below shows sample inputs without noise. The voltage swell (Sw) disturbance signal appears at the top, followed by the voltage sag signal (Sg). The signals at the bottom pertain to disturbances with harmonics (SwH and SgH).

Results

The classification algorithm was implemented in Matlab™. The raw data was classified into training and test samples. For each category, 70% were retained as training samples and the rest was used for testing the overall performance of the classification method. Next, the V-detectors algorithm was run for a total of 1,000 iterations to generate Nonself detectors. The radii around the Self samples was $\gamma = 0.1$. Each category detector was subject to proliferation where the unit directions were $\hat{u} = [0 \ 0 \pm 1]$, $[0 \pm 1 \ 0]$ and $[\pm 1 \ 0 \ 0]$. The reachability coefficient during the proliferation stages was maintained at a constant value of $\eta = 2.0$.

The results of this algorithm are shown in Table 1. Each row corresponds to the outcome when test samples from that single category were presented to the algorithm. Each column indicates how the algorithm classified the test sample. When any input test sample is not detected by detectors belonging to any category, it is automatically classified as Nonself, which is shown in the rightmost column. For example, when Sg samples were presented to the algorithm, 92% were correctly identified as Sg, 4%

Table 1. Final classification results of proposed method.

	Sw	Sg	SwH	SgH	N
Sw	73%	0	27%	0	0
Sg	0	92%	4%	0	4%
SwH	27%	0	67%	6%	0
SgH	0	0	0	100%	0

were identified as SwH and 4% were not recognized. The overall accuracy – the proportion of total samples that were correctly classified – was 84.29%. The accuracy can be increased by increasing the size of the training data set, and/or by adjusting the Self-radius of the Self samples.

CONCLUSION

This chapter introduced anomaly detection methods analogous to methods employed by the vertebrate immune system. In summary, the immune system works by negative selection, which is superior to positive selection in its ability to react to unknown threats. Algorithms based on negative selection create detectors which, analogous to antibodies, are designed to detect anomalous inputs. In computer applications, the inputs and the detectors are usually represented with binary strings. The detection mechanism basically consists of comparing bits in the input string with bits in the detector strings and classifying inputs based on match/mismatch. In engineering applications, the inputs and the detectors are usually real-valued vectors, and the detection mechanism typically consists of determining if any detector in the antibody repertoire detects an observed input.

The basic negative selection algorithm, applied to real-valued signals, was described in great detail in this chapter. First, the basic negative selection algorithm was outlined. Then the focus was shifted to a subclass of detectors, called V-detectors, or variable-sized detectors, which are useful in many engineering applications. Then several methods of improving the negative selection algorithm were suggested. Specifically, this chapter proposed a way to categorize all such existing schemes into three basic categories: Self-organizing detectors, evolving detectors and proliferating detectors. Lastly, the proliferation mechanism was described in detail, and extending the detector proliferation method beyond anomaly detection to also perform multi-category classification was proposed.

Two case studies were used to demonstrate the effectiveness of algorithms based on the immune system. First, the V-detector and proliferating V-detectors algorithms were applied to a real world engineering problem of detecting anomalous data for an automated bearing testing machine. Then, the proliferating V-detectors algorithm was used to solve a power quality disturbance classification problem. The simulation results verified the effectiveness of the algorithms.

REFERENCES

- Aickelin, U., & Cayzer, S. (2002). The danger theory and its application to artificial immune systems. In J. Timmis & P.J. Bentley (Eds.), *Proceedings of the International Conference on Artificial Immune Systems (ICARIS)* (pp. 141-148). University of Kent at Canterbury.
- Aickelin, U., Greensmith, J., & Twycross, J. (2004). Immune system approaches to intrusion detection - a review. In *Proceedings of the 3rd International Conference on Artificial Immune Systems* (LNCS 3239, pp. 316-329). Berlin, Germany: Springer.
- Balachandran, S., Dasgupta, D., Nino, F., & Garrett, D. (2007). A framework for evolving multi-shaped detectors in negative selection. In *Proceedings of the IEEE Symposium on Foundations of Computational Intelligence*, Honolulu, Hawaii (pp. 401-408).

- Branco, P. J. C., Dente, J. A., & Mendes, R. V. (2003). Using immunology principles for fault detection. *IEEE Transactions on Industrial Electronics*, 50(2), 362–373. doi:10.1109/TIE.2003.809418
- Das, S., Gui, M., & Pahwa, A. (2008). Artificial immune systems for self-nonsel discrimination: Application to anomaly detection. *Advances of Computational Intelligence in Industrial Systems*, 116, 231–248. doi:10.1007/978-3-540-78297-1_11
- Das, S., Natarajan, B., Stevens, D., & Koduru, P. (2008). Multi-objective and constrained optimization for DS-CDMA code design based on the clonal selection principle. *Applied Soft Computing Journal*, 8, 788–797. doi:10.1016/j.asoc.2007.05.012
- Das, S., & Panigrahi, P. K. (2008). Evolutionary algorithms for multi-objective optimization. In J. Dopico, J. de la Calle, & A. Sierra (Eds.), *Encyclopedia of artificial intelligence* (pp. 1145-1151). Hershey, PA: Information Science Reference.
- Dasgupta, D., Krishna-Kumar, K., Wong, D., & Berry, M. (2004). Negative selection algorithm for aircraft fault detection. In *Artificial immune systems*, (LNCS 3239, pp. 1-13). Berlin, Germany: Springer.
- Dasgupta, D., & Gonzalez, F. (2002). An immunity-based technique to characterize intrusions in computer networks. *IEEE Transactions on Evolutionary Computation*, 6(3), 281–291. doi:10.1109/TEVC.2002.1011541
- Dasgupta, D., & Gonzalez, F. (2003). Anomaly detection using real-valued negative selection. *Journal of Genetic Programming and Evolvable Machines*, 4(4), 383–403. doi:10.1023/A:1026195112518
- Dasgupta, D., Yu, S., & Majumdar, N. (2005). MILA -- multi-level immune learning algorithm and its application to anomaly detection. *The Soft Computing Journal*, 9(3), 172–184. doi:10.1007/s00500-003-0342-7
- Dash, P. K., Panigrahi, B. K., & Panda, G. (2003). Power quality analysis using S-transform. *IEEE Transactions on Power Delivery*, 18(2), 406–411. doi:10.1109/TPWRD.2003.809616
- De Castro, L. N., & Timmis, J. (2002). *Artificial immune systems: A new computational intelligence approach*. London: Springer-Verlag.
- De Castro, L. N., & Timmis, J. (2003). Artificial immune systems as a new soft computing paradigm. *Soft Computing - A Fusion of Foundations . Methodologies and Applications*, 7(8), 526–544.
- de Lemos, R., Timmis, J., Ayara, M., & Forrest, S. (2007). Immune-inspired adaptable error detection for automated teller machines. *IEEE Transactions on Systems, Man and Cybernetics. Part C, Applications and Reviews*, 37(5), 873–886. doi:10.1109/TSMCC.2007.900662
- Gao, X.-Z., Ovaska, S. J., & Wang, X. (2008). A GA based negative selection algorithm. *International Journal of Innovative Computing . Information and Control*, 4(4), 971–979.
- Gui, M., Das, S., & Pahwa, A. (2007). Procreating v-detectors for nonsel recognition: An application to anomaly detection in power systems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, London, UK (pp. 261-268).

- Harmer, P. K., Williams, P. D., Gunsch, G. H., & Lamont, G. B. (2002). An artificial immune system architecture for computer security applications. *IEEE Transactions on Evolutionary Computation*, 6(3), 252–280. doi:10.1109/TEVC.2002.1011540
- Ji, Z. (2005). A boundary-aware negative selection algorithm. In *Proceedings of the 9th IASTED International conference on Artificial intelligence and soft computing (ASC'05)*, Benidorm, Spain.
- Ji, Z., & Dasgupta, D. (2004). Real valued negative selection algorithm using variable sized detectors. In *Proceedings of the Genetic and Evolutionary Computation Conference*, Seattle, WA (pp. 287-298).
- Ji, Z., & Dasgupta, D. (2006). Applicability issues of the real-valued negative selection algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, Seattle, WA (pp. 111-118).
- Ji, Z., & Dasgupta, D. (2007). Revisiting negative selection algorithms. *Evolutionary Computation Journal*, 15(2), 223–251. doi:10.1162/evco.2007.15.2.223
- Luh, G. C., & Chen, W. (2005). Immune model-based fault diagnosis. *Mathematics and Computers in Simulation*, 67(6), 515–539. doi:10.1016/j.matcom.2004.07.004
- Nio, F., Gomez, D., Wong, D., & Vejar, R. (2003). A novel immune anomaly detection technique based on negative selection. In *Artificial immune systems* (LNCS 2723, p. 204). Berlin, Germany: Springer.
- Nunn, I., & White, T. (2005). The application of antigenic search techniques to time series forecasting. In *Proceedings of the Genetic and Evolutionary Computation Conference*, Washington, DC (pp. 353-360).
- Sahai, S., & Pahwa, A. (2004). Failures of overhead distribution system lines due to animals. In *Proceedings of North American Power Symposium*, Moscow, Idaho.
- Sarafijanovic, S., & Le Boudec, J.-Y. (2005). An artificial immune system approach with secondary response for misbehavior detection in mobile ad hoc networks. *IEEE Transactions on Neural Networks*, 16(5), 1076–1087. doi:10.1109/TNN.2005.853419
- Stockwell, R. G., Mansinha, L., & Lowe, R. P. (1996). Localization of the complex spectrum: The S-transform. *IEEE Transactions on Signal Processing*, 44(4), 998–1001. doi:10.1109/78.492555
- Strackeljan, J., & Leiviskä, K. (2008). Artificial immune system approach for the fault detection in rotating machinery. In Proceedings of the International Conference on Condition Monitoring & Machinery Failure Prevention Technologies, Edinburgh, UK.
- Taylor, D. W., & Corne, D. W. (2003). An investigation of the negative selection algorithm for fault detection in refrigeration systems. In *Artificial immune systems* (LNCS 2787, pp. 34-45). Berlin, Germany: Springer.

KEY TERMS AND DEFINITIONS

Affinity: the ability of different antibodies to bind to pathogens.

AIS: Artificial Immune System, a broad range of algorithms that are motivated by the vertebrate immune system

Anomaly Detection: a process to determine if a variable is within the normal region of operation.

Antibody: a cell which has the ability to bind to pathogens.

Artificial Immune System: an algorithm based on the functionality of the vertebrate immune system.

Biologically Inspired Algorithm: a class of probabilistic algorithms based on biological metaphors and widely used in complex problem solving.

Evolutionary Algorithm: an algorithm based on the Darwinian theory of evolution.

Detector: the equivalent of an antibody in an artificial immune system.

Fitness: a function characterising the comparative strengths / benefits of a solution.

Hypersphere: an equivalent of a sphere with more than 3 dimensions.

Limit Plane: a hyperplane with a single fixed dimension value, corresponding to the upper or lower limits of a variable

Negative Characterisation: an identification of Nonself based on known samples of Self.

Nonself: cells foreign to the body (abnormal region of operation).

Pathogens: foreign, potentially harmful cells

Positive Characterisation: an identification of Nonself based on known samples of Nonself.

Self: a body's own cells (normal region of operation).

Self-Nonself Discrimination: the ability of antibodies to distinguish between Self and Nonself.

Chapter 6

Calibration of Machine Learning Models

Antonio Bella

Universidad Politécnica de Valencia, Spain

Cèsar Ferri

Universidad Politécnica de Valencia, Spain

José Hernández-Orallo

Universidad Politécnica de Valencia, Spain

María José Ramírez-Quintana

Universidad Politécnica de Valencia, Spain

ABSTRACT

The evaluation of machine learning models is a crucial step before their application because it is essential to assess how well a model will behave for every single case. In many real applications, not only is it important to know the “total” or the “average” error of the model, it is also important to know how this error is distributed and how well confidence or probability estimations are made. Many current machine learning techniques are good in overall results but have a bad distribution assessment of the error. For these cases, calibration techniques have been developed as postprocessing techniques in order to improve the probability estimation or the error distribution of an existing model. This chapter presents the most common calibration techniques and calibration measures. Both classification and regression are covered, and a taxonomy of calibration techniques is established. Special attention is given to probabilistic classifier calibration.

INTRODUCTION

One of the main goals of machine learning methods is to build a model or hypothesis from a set of data (also called evidence). After this learning process, the quality of the hypothesis must be evaluated as precisely as possible. For instance, if prediction errors have negative consequences in a certain application domain of a model (for example, detection of carcinogenic cells), it is important to know the exact

DOI: 10.4018/978-1-60566-766-9.ch006

accuracy of the model. Therefore, the model evaluation stage is crucial for the real application of machine learning techniques. Generally, the quality of predictive models is evaluated by using a training set and a test set (which are usually obtained by partitioning the evidence into two disjoint sets) or by using some kind of cross-validation or bootstrap if more reliable estimations are desired. These evaluation methods work for any kind of estimation measure. It is important to note that different measures can be used depending on the model. For classification models, the most common measures are accuracy (the inverse of error), f-measure, or macro-average. In probabilistic classification, besides the percentage of correctly classified instances, other measures such as logloss, mean squared error (MSE) (or Brier's score) or area under the ROC curve (AUC) are used. For regression models, the most common measures are MSE, the mean absolute error (MAE), or the correlation coefficient.

With the same result for a quality metric (e.g. MAE), two different models might have a different error distribution. For instance, a regression model R_1 that always predicts the true value plus 1 has a MAE of 1. However, it is different to a model R_2 that predicts the true value for $n - 1$ examples and has an error of n for one example. Model R_1 seems to be more reliable or stable, i.e., its error is more predictable. Similarly, two different models might have a different error assessment with the same result for a quality metric (e.g. accuracy). For instance, a classification model C_1 which is correct 90% of the cases with a confidence of 0.91 for every prediction is preferable to model C_2 which is correct 90% of the cases with a confidence of 0.99 for every prediction. The error self-assessment, i.e., the purported confidence, is more accurate in C_1 than in C_2 .

In both cases (classification and regression), an overall picture of the empirical results is helpful in order to improve the reliability or confidence of the models. In the case of regression, the model R_1 , which always predicts the true value plus 1, is clearly uncalibrated, since predictions are usually 1 unit above the real value. By subtracting 1 unit from all the predictions, R_1 could be calibrated and interestingly, R_2 can be calibrated in the same way. In the case of classification, a global calibration requires the confidence estimation to be around 0.9 since the models are right 90% of the time.

Thus, calibration can be understood in many ways, but it is usually built around two related issues: how error is distributed and how self-assessment (confidence or probability estimation) is performed. Even though both ideas can be applied to both regression and classification, this chapter focuses on error distribution for regression and self-assessment for classification.

Estimating probabilities or confidence values is crucial in many real applications. For example, if probabilities are accurate, decisions with a good assessment of risks and costs can be made using utility models or other techniques from decision making. Additionally, the integration of these techniques with other models (e.g. multiclassifiers) or with previous knowledge becomes more robust. In classification, probabilities can be understood as degrees of confidence, especially in binary classification, thus accompanying every prediction with a reliability score (DeGroot & Fienberg, 1982). In regression, predictions might be accompanied by confidence intervals or by probability density functions.

Therefore, instead of redesigning existing methods to directly obtain good probabilities or better error distribution, several calibration techniques have recently been developed. A calibration technique is any postprocessing technique that attempts to improve the probability estimation or to improve the error distribution of a given predictive model. A general calibration technique can be used to improve any existing machine learning method: decision trees, neural networks, kernel methods, instance-based methods, Bayesian methods, etc. It can also be applied to hand-made models, expert systems or combined models.

Depending on the task, different calibration techniques can be applied and the definition of calibration can be stated more precisely. The most common calibration techniques are divided into four groups and each group has a type code to identify it:

- TYPE CD. Calibration techniques for discrete classification (“(class) distribution calibration in classification” or simply “class calibration”): a typical decalibration arises when the model predicts examples of one or more classes in a proportion that does not fit the original class distribution. In the binary case (two classes), it can be expressed as a mismatch between the expected value of the proportion of classes and the actual one. For instance, if a problem has a proportion of 95% of class a and 5% of class b , a model predicting 99% of class a and 1% of class b is uncalibrated, although it could have a low error (ranging from 4% to 5%). This error distribution can be clearly shown on a confusion or contingency table. Therefore, class calibration is defined as the degree of approximation of the true or empirical class distribution with the estimated class distribution. The standard calibration model procedure is performed by changing the threshold that determines when the model predicts a or b , making this threshold stricter with class a and milder with class b to balance the proportion. Note that, in principle, this type of calibration might produce more error. In fact, this is usually the case when a useful model for problems with very imbalanced class distribution must be obtained, i.e., the minority class has very few examples. Note that this calibration can be used to match global proportions as well as local proportions. This is related to the problem of “repairing concavities” (Flach & Wu, 2005).
- TYPE CP. Calibration techniques for probabilistic classification (“probabilistic calibration for classification”): a probabilistic classifier is a decision system that accompanies each prediction with a probability estimation. If the classification model predicts that it is 99% sure, it should expect to be right 99% of the time. If it is only right 50% of the time, it is not calibrated because its estimation was too optimistic. Similarly, if it predicts that it is only 60% sure, it should be right 60% of the time. If the classifier is right 80% of the time, it is not calibrated because its estimation was too pessimistic. In both cases, the expected value of the number or proportion of correct guesses (in this case, the probability or the confidence assessment) does not match the actual value. Calibration is thus defined as the degree of approximation of the predicted probabilities to the actual probabilities. More precisely, a classifier is perfectly calibrated if, for a sample of examples with predicted probability p , the expected proportion of positives is close to p . Note that even though, accuracy and calibration are dependent on each other, they are very different things. For instance, a random classifier (a coin tosser), which always assigns 0.5 probability to its predictions, is perfectly calibrated. On the other hand, a very good classifier can be uncalibrated if correct positive (respectively negative) predictions are accompanied by relatively low (respectively high) probabilities. Also note that good calibration usually implies that estimated probabilities are different for each example (except for the random coin tosser). For some examples, confidence will be high, and for other more difficult ones, confidence will be low. This implies that measures to evaluate this type of calibration must evaluate agreement between the expected value and the real value in a local way by using partitions or bins of the data.
- TYPE RD. Calibration techniques to fix error distribution for regression (“distribution calibration in regression”): the errors in this case are not regularly distributed along the output values. The error is concentrated in the high values, or the average error (either positive or negative) is not close to zero. The expected value, which should be close to the actual value, can be defined

in several ways. For instance, the expected value of the estimated value (y_{est}) should be equal (or close) to the real value (y), i.e., $E(y_{est}) = E(y)$ or, equivalently, $E(y_{est} - y) = 0$. In the example R_1 above, $E(y_{est}) = E(y) + 1$. The mean error (its expected value) would be 1 and not 0. Another equation that shows that a model might be uncalibrated is the expected value of the quotient between the estimated value and the real value, $E(y_{est} / y)$, which should be equal or close to 1. If this quotient is greater than one, the error is usually positive for high values and negative for low values. Typically, these problems are detected and penalised by classical measures for evaluating regression models (Montgomery, Peck, & Vining, 2001). Also, many techniques (e.g. linear regression) generate calibrated models (at least for error distribution and self-assessment). Other kinds of more sophisticated techniques especially hand-made models might be uncalibrated and may require a calibration.

- TYPE RP. Calibration techniques to improve probability estimation for regression (“probabilistic calibration for regression”): this is a relatively new area (Carney & Cunningham, 2006) and is applicable when continuous predictions are accompanied or substituted by a probability density function (or, more restrictively, confidence intervals). Regression models of this kind are usually referred to as “density forecasting” models. Instead of saying that the temperature is going to be 23.2° Celsius, a probability density function can be given assigning a probability of 0.9 indicating that the temperature is between 21° and 25° is 0.9, or a probability of 0.99 indicating that the temperature is between 15° and 31°. If the predictions are very accurate, the density functions (and, hence, confidence intervals) should be narrower. If the predictions are bad, the density functions should be broader in order to approximate the estimated probabilities to the real probabilities. As in the type CP, in general, a good calibration requires that these density functions be specific to each prediction, i.e., for some cases where the confidence is high, confidence intervals will be narrower. For difficult cases, confidence intervals will be broader. As in the type CP, measures to evaluate this type of calibration must evaluate agreement between the expected value and the real value in a local way by using partitions or bins of the data.

Table 1 summarises these four types of calibration.

Note that types CD and RD must necessarily modify predictions in order to calibrate the results. In type CD, if the class threshold is moved, some predictions change. In RD if an attempt is made to reduce high values and increase low values, predictions also change. In contrast, for types CP and RP, calibration can be made without having to modify predictions: only confidence assessments or probabilities need to be adjusted. For CP, in particular, these kinds of calibrations are known as *isotonic*. Consequently, some measures such as average error will not be affected by these two types of calibrations.

Additionally, if calibration is to be improved, measures are also needed to evaluate this improvement. A calibration measure is any measure that is able to quantify the degree of calibration of a predictive model. For each type of calibration model, some specific measures are useful to evaluate the degree of calibration, while others are only partially sensitive or completely useless. For instance, for CP, the most common measure, accuracy (or % of errors), is completely useless. For RP, the most common measure, MSE, is completely useless. Some of these calibration measures are reviewed in the following section.

Of all the types shown in Table 1, type CP is the one that has recently received the most attention. In fact, for many researchers in machine learning, the term “calibration” usually refers to this type, without having to specify that there are other types of calibration. Additionally, this is the type that has

Table 1. A taxonomy of calibration problems

TYPE	Task	Problem	Global/Local	What is calibrated?
CD	Classification	Expected class distribution is different from real class distribution	Global or local	Predictions
CP	Classification	Expected/estimated probability of correct guesses is different from the real proportion.	Local	Probability/confidence
RD	Regression	Expected output is different from the real average output.	Global or local	Predictions
RP	Regression	Expected/estimated error confidence intervals or probability density functions are too narrow or too broad.	Local	Probability/confidence

developed more techniques and more specific measures. Furthermore, regression techniques and measures have been traditionally developed to obtain calibrated models, so less improvement is expected from calibration techniques. For this reason, this chapter focuses mainly on classification problems, and specifically on type CP.

This chapter provides a general overview of calibration and a review of some of the most-well-known calibration evaluation measures and calibration methods which have been proposed for classification and regression. This chapter also analyses some open questions and challenges that affect future research on calibration.

CALIBRATION EVALUATION MEASURES

As mentioned in the introduction, a calibration measure is any measure that can quantify the degree of calibration of a predictive model. As Table 2 shows, many classical quality metrics are not useful for evaluating calibration techniques. In fact, new and specific measures have been derived or adapted to evaluate calibration, especially for types CP and RP.

The second column in Table 2 shows the calibration measures. However, does not mean that these measures *only* measure calibration. For instance, for type CP, CalBin and CalLoss only evaluate calibration, while MSE or Logloss evaluate calibration as well as other components. These two types of measures are referred to as *pure* and *impure* calibration measures, respectively. However, pure calibration measures may make a classifier which always predicts the positive prior probability look perfectly calibrated according to these measures. Other impure metrics are insensitive to calibration: for example, qualitative measures (accuracy, mean F-measure, etc.), where the calibration function is applied to the threshold; or ranking measures (such as AUC), where the calibration modifies the value of the probabilities but not their order. Therefore, calibration has emerged as an important issue, since many traditional quality metrics completely disregard it. Hence, many machine learning techniques generate uncalibrated models.

Note that we use two different terms for Mean Square Error, MSE^p and MSE^r . The reason for this is that MSE^p is used for classification and compares the estimated probabilities with the actual probability (0 or 1), while MSE^r is used for regression and compares two continuous values.

Most of the measures shown in the second column of Table 2 are very well-known and do not require any further definition. Nevertheless, it is important to remark the following: macro-averaged accuracy

Table 2. The second column shows the calibration measures for each type of calibration problem. References and definitions for all the measures can be found in (Ferri, Hernández-Orallo, & Modroiu, 2008; Carney & Cunningham, 2006). The third and fourth columns show measures which are partially sensitive (but not very useful in general) or completely insensitive to each type of calibration.

TYPE	Calibration measures	Partially sensitive measures	Insensitive measures
CD	Macro-averaged accuracy, proportion of classes.	Accuracy, mean F-measure, ...	Area Under the ROC Curve (AUC), MSE^p , Logloss, ...
CP	MSE^p , LogLoss, CalBin, CalLoss		AUC, Accuracy, mean F-measure, ...
RD	Average error, Relative error	MSE^r , MAE, ...	
RP	Anderson-Darling (A2) test		Average error, relative error, MSE^r , MAE, ...

is the average of the partial accuracies for each class; the proportion of classes is computed for the predictions on a dataset and can be compared with the real proportion; and average error and relative error are well-known in regression. The rest of this section is devoted to explaining MSE^p , LogLoss, CalBin, CalLoss for type CP and Anderson-Darling (A2) test for RP.

Calibration Measures for Type CP

In this chapter, the following notation is used. Given a dataset T , n denotes the number of examples, and C denotes the number of classes. $f(i, j)$ represents the actual probability of example i to be of class j . It is assumed that $f(i, j)$ always takes values in $\{0, 1\}$ and is strictly not a probability but an indicator function.

The number of examples of class j is denoted as $n_j = \sum_{i=1}^n f(i, j)$. $p(j)$ denotes the prior probability of class j , i.e., $p(j) = n_j / n$. Given a classifier, $p(i, j)$ represents the estimated probability of example i to be of class j taking values in $[0, 1]$.

Mean Squared Error

Mean Squared Error (MSE) is a measure of the deviation from the true probability. In classification, MSE^p is also known as Brier Score.

MSE is defined as:

$$MSE = \sum_{j=1}^C \sum_{i=1}^n \frac{(f(i, j) - p(i, j))^2}{n \times C}, \quad (1)$$

Although MSE was originally not a calibration measure, it was decomposed in (Murphy, 1972) in terms of calibration loss and refinement loss. An important idea of decomposition is that data is organised into bins, and the observed probability in each bin is compared to the predicted probability or to the global probability. Some kind of binning is present in many calibration methods and measures. For decomposition, T is segmented in k bins.

$$MSE = \frac{\sum_{j=1}^C \sum_{l=1}^k \sum_{i=1, i \in l}^{n_l} n_l \times (p(i, j) - \bar{f}(i, j))^2 - \sum_{l=1}^k n_l \times (\bar{f}_l(i, j) - \bar{f}(j)) + \bar{f}(j) \times (1 - \bar{f}(j))}{n \times C}, \quad (2)$$

where $\bar{f}_l(i, j) = \sum_{i=1, i \in l}^{n_l} \frac{f(i, j)}{n_l}$ and $\bar{f}(j) = \sum_{i=1}^n \frac{f(i, j)}{n}$. The first term measures the calibration of the classifier while the rest of the expression measures other components, which are usually grouped under the term “refinement”. The calibration component is the only one that is affected by isotonic calibrations. The other components, discrimination and uncertainty, are not modified if probabilities are calibrated in such a way that the order is not modified (i.e., isotonic), since bins will not be altered.

LogLoss

Logloss is a similar measure and is also known as probabilistic cost or entropy. It is related to the Kullback-Leibler distance between the real model and the inferred model (Good, 1952; Good, 1968; Dowe, Farr, Hurst, & Lentin, 1996). It is defined as follows:

$$Logloss = \sum_{j=1}^C \sum_{i=1}^n \frac{(f(i, j) \times \log p(i, j))}{n}, \quad (3)$$

Calibration by Overlapping Bins

One typical way of measuring classifier calibration consists of splitting the test set into several segments or bins, as the MSE decomposition shows (even though MSE does not need to use bins to be computed). The problem of using bins is that if too few bins are defined, the real probabilities are not properly detailed to give an accurate evaluation. Also, if too many bins are defined, the real probabilities are not properly estimated. A partial solution to this problem is to make the bins overlap.

A calibration measure based on overlapping binning is CalBin (Caruana & Niculescu-Mizil, 2004). This is defined as follows: for each class, all cases must be ordered by predicted $p(i, j)$, giving new indices i^* . Then, the first 100 elements (i^* from 1 to 100) are taken as the first bin. Then, the percentage of cases of class j in this bin is calculated as the actual probability, \hat{f}_j . The error for this bin is $\sum_{i^* \in 1..100} |p(i^*, j) - \hat{f}_j|$. The second bin with elements (2 to 101) is taken and its error is computed in the same way. Finally, the errors are averaged. The problem of using 100 as the size of each bin (as Caruana and Niculescu-Mizil (2004) suggest) is that it might be too large of a bin for small datasets. Instead of 100, a different bin length, $s = n/10$ could be set in order to make it more size-independent. Formally:

$$CalBin(j) = \frac{1}{n-s} \sum_{b=1}^{n-s} \sum_{i^*=b}^{b+s-1} \left| p(i^*, j) - \frac{\sum_{i^*=b}^{b+s-1} f(i^*, j)}{s} \right|, \quad (4)$$

Calibration Loss

Calibration can clarify the relationship between the AUC-based measures and ROC analysis (Fawcett & Niculescu-Mizil, 2007; Flach & Matsubara, 2007). The receiver operating characteristic curve (ROC curve) is a graphical depiction of classifiers based on their performance. It is generally applied to binary classifiers. The ROC space is a two-dimensional graph in which the True positive rate (the fraction of positives correctly classified or *tp rate*) is plotted on the Y axis, and the False positive rate (the fraction of negatives incorrectly classified or *fp rate*) is plotted on the X axis. Each discrete classifier produces an (*fp rate*, *tp rate*) pair that corresponds to a single point in the ROC space. Probabilistic classifiers provide a value (probability or score) that represents the degree to which an instance belongs to a class. In combination with a threshold, the classifier can behave as a binary classifier by assigning a class (for instance, positive) if the produced score is above the threshold or by assigning the other class (negative) otherwise. Each threshold produces a point in the ROC space, and a ROC curve is generated by drawing a line crossing all the points. The area under a ROC curve is abbreviated as AUC. “The AUC has an important statistical property: the AUC of a classifier is equivalent to the probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance. This is equivalent to the Wilcoxon test of ranks” (Fawcett, 2006, p. 868). Therefore, the AUC is a class separability or instance ranking measure because it evaluates how well a classifier ranks its predictions.

A perfectly calibrated classifier always gives a convex ROC curve. However even though, a classifier can produce very good rankings (high AUC), the estimated probabilities might differ from the actual probabilities.

One method for calibrating a classifier is to compute the convex hull or, equivalently, to use isotonic regression. From the decomposition of the Brier Score, Flach and Matsubara (2007) derive calibration loss and refinement loss, where calibration loss is defined as the mean squared deviation from empirical probabilities that are derived from the slope of ROC segments.

$$CalLoss(j) = \sum_{b=1}^{r_j} \sum_{i \in s_{j,b}} \left(p(i,j) - \sum_{i \in s_{j,b}} \frac{f(i,j)}{|s_{j,b}|} \right)^2, \quad (5)$$

where r_j is the number of segments in the ROC curve for class j , i.e. the number of different estimated probabilities for class j : $|\{p(i,j)\}|$. Each ROC segment is denoted by $s_{j,b}$, with $b \in 1 \dots r_j$, and formally defined as:

$$s_{j,b} = \left\{ i \in 1 \dots n \mid \forall k \in 1 \dots n : p(i,j) \geq p(k,j) \wedge i \notin s_{j,d}, \forall d < b \right\}, \quad (6)$$

Calibration Measures for Type RP

Anderson-Darling (A^2) Test

For type RP, where the task is usually referred to as density forecasting, instead of predicting a continuous value, the prediction is a probability density function. Evaluating this probability density function

Table 3.

Predicted		Real		
	<i>a</i>	<i>b</i>	<i>c</i>	
<i>a</i>	20	2	3	
<i>b</i>	0	30	3	
<i>c</i>	0	2	40	

in terms of calibration cannot be done with classical measures such as MSE' , relative quadratic error, or other classical measures in regression. Carney and Cunningham (2006) adapt a well-known normality test, the Anderson-Darling (A^2) test over the probability integral transformation, as a measure of pure calibration. This measure is used to evaluate whether the probability density functions estimated by the regression model are accurate. For the specific definition see Carney & Cunningham (2006).

Calibration Methods for Type CD

In the case of discrete classification, the best way to know whether a model is uncalibrated according to the number of instances per class (type CD) is to analyse the confusion matrix. The confusion matrix is a visual way of showing the recount of cases of the predicted classes and their actual values. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class. One benefit of a confusion matrix is that it is easy to see if the system is confusing two classes (i.e., commonly mislabelling one as another). For example, if there are 100 test examples and a classifier, an example of a confusion matrix with three classes *a*, *b* and *c* could be as shown in Table 3.

In this matrix from 100 examples: 20 were from class *a* and all of them were well classified; 34 were from class *b*, and 30 of them were well classified as *b*, 2 were misclassified as *a*, and 2 were misclassified as *c*; finally, 46 of the examples were from class *c*, 40 of them were well classified as *c*, 3 were misclassified as *a* and 3 were misclassified as *b*. If they are grouped by class, there is a proportion of 20, 34, 46 for the real data, and a proportion of 25, 33, 42 for the predicted data. These proportions are quite similar and, therefore, the classifier is calibrated with regard to the original class distribution.

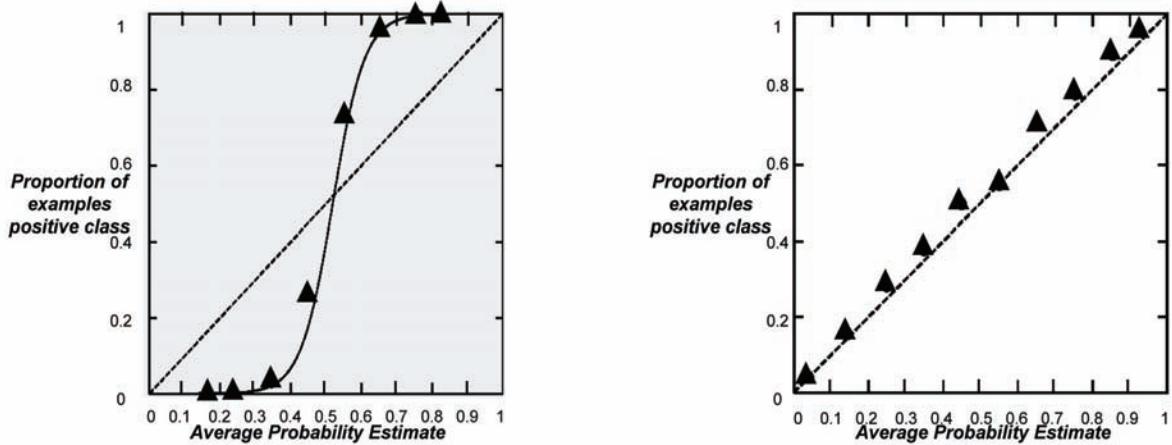
Another matrix can also be considered (Table 4).

In this matrix, the proportion of real data for classes *a* and *b* are (100, 25), while the proportion of predicted data are (62, 63). Thus, in this case, the model is uncalibrated. This type of disproportion is quite common and usually favours the majority classes. There are techniques to solve the problem once the model is obtained. To do so, the predictions of the models must be accompanied by probabilities or reliability values (or simply, scores). Then, the threshold that splits into the classes can be changed.

Table 4.

Predicted		Real	
	<i>a</i>	<i>b</i>	
<i>a</i>	60	2	
<i>b</i>	40	23	

Figure 1. Left: an example of a reliability diagram of an uncalibrated model. Right: an example of a reliability diagram of a calibrated model.



The technique presented by (Lachiche & Flach, 2003) can be applied. Even though it is specialised in Bayesian classifiers, it can be applied to any classification model that accompanies its predictions by probabilities or reliabilities. A naïve Bayes classifier estimates the probabilities for each class independently. For example, for the following probabilities for each class: $p(a|x) = 0.2$ and $p(b|x) = 0.05$, there are no more classes. Therefore, the sum of the probabilities is not 1. In general, the naïve Bayes classifiers assign very low probabilities because the probability is the product of several factors that, in the end, reduce the absolute values of the probabilities. The decision rule used to apply the model is:

If $p(a|x) > p(b|x)$ then *a*

else *b*

The consequence of this rule is that the result is not calibrated in most of the cases. It may be that this rule produces many more examples of the class *a* (or *b*) than there were in the original distribution. The solution to this problem is to estimate a threshold that is fitted to the original distribution.

If there are only two classes, the solution is very easy. A ratio of the two probabilities can be calculated: $r = p(a|x)/p(b|x)$. This ratio is 0 to infinite. It can also be normalised between 0 and 1 with a sigmoid, if desired. The aim is to obtain a threshold u with the test set where the distribution of the model is similar to the original distribution. Thus, the rule changes to:

If $r > u$ then *a*

else *b*

Lachiche and Flach (2003) show that the results of the models can be improved significantly with only a small adjustment (in that work the threshold adjustment is based on the ROC analysis and it is extended to multiclass). In particular, from 25 analysed datasets, this simple optimisation significantly improved the accuracy in 10 cases and was reduced in only 4 of them.

Apart from that simple approximation, there are other works that calculate the optimum threshold fitted to the original distribution, such as (O'Brien, Gupta, & Gray, 2008).

Calibration Methods for Type CP

Another case is the calibration of probabilistic classification models (type CP), which requires more sophisticated techniques. In this case, when the model predicts that the probability of the class a is 0.99, this means that the model is more confident that the class is a than when the probability is 0.6. Determining the reliability of a prediction is fundamental in many applications such as: diagnosis, instance selection, and model combination.

In addition to the measures introduced above (MSE, logloss, CalBin and CalLoss), the fundamental tool for analysing the calibration of models of this type is the reliability diagram (DeGroot & Fienberg, 1982). In this diagram, the prediction space is discretised into 10 intervals (from 0 to 0.1, from 0.1 to 0.2, etc.). The examples whose probability is between 0 and 0.1 go into the first interval, the examples between 0.1 and 0.2 go into the second, etc. For each interval, the mean predicted value (in other words, the mean predicted probability) is plotted (x axis) versus the fraction of positive real cases (y axis). If the model is calibrated, the points will approach the diagonal.

Figure 1 shows an example of an uncalibrated model and a calibrated model.

For instance, in the model on the left, there are no examples with a predicted probability lower than 0.1. The next interval, where the examples have an assigned probability between 0.1 and 0.2 for the positive class (with a mean of 0.17), there are no examples of the positive class. Thus, these predictions have an estimated probability that is too high. It should be closer to 0 instead of to 0.17. The values at the end of the curve show that the examples with assigned probabilities between 0.7 and 0.8 are all from the positive class. They should probably have a higher probability because they are cases with greater certainty.

The model on the right shows that the correspondence is more correct: there are probabilities distributed from 0 to 1 and, moreover, they are usually the same as the percentage of examples.

There are several techniques that can calibrate a model like the one on the left and transform it into a model like the one on the right. The most common are: binning averaging, isotonic regression and Platt's method. The objective of these methods (as postprocessing) is to transform the original estimated probabilities¹ $p(i, j)$ into calibrated probability estimates $p^*(i, j)$. It is important to remark that all of these general calibration methods can only be used (directly, without approaches) in binary problems because they all use the sorted estimated probability to calculate the calibrated probability.

The calibration function is monotonically non-decreasing (also called isotonic). Most calibration methods presented in the literature are isotonic. This makes it reasonable to use MSE or LogLoss as measures to evaluate calibration methods since the “separability components” are not affected. This is clearly seen in the so-called “decompositions of the Brier score” (Sanders, 1963; Murphy, 1972), which is described above.

Binning Averaging

The first calibration method is called binning averaging (Zadrozny & Elkan, 2001) and consists of sorting the examples in decreasing order by their estimated probabilities and dividing the set into k bins (i.e., subsets of equal size). Then, for each bin b , $1 \leq b \leq k$, the corrected probability estimate for a case i

belonging to class j , $p^*(i, j)$, is the proportion of instances in b of class j . The number of bins must be small in order to reduce the variance of the estimates. In their paper, Zadrozny and Elkan fixed $k=10$ in the experimental evaluation of the method.

An illustrative example for explaining how this method works is shown in Table 5. Consider the following training set sorted by its probability of membership to the positive class grouped in 5 bins.

Then, if a new example is assigned a score of 0.68, it belongs to bin 3 and its calibrated probability is $\frac{0.70 + 0.66 + 0.62 + 0.62}{4} = 0.65$.

Isotonic Regression (Pair-Adjacent Violator Algorithm, PAV)

Another slightly more sophisticated technique also for two-class problems is isotonic regression. (Ayer, Brunk, Ewing, Reid, & Silverman, 1955) presented a pair-adjacent violator algorithm (PAV) for calculating isotonic regression. The idea is that calibrated probability estimates must be a monotone decreasing sequence, i.e., $p_1 \geq p_2 \geq \dots \geq p_n$. If the sequence is not satisfied each time that a pair of consecutive probabilities, $p(i, j)$ and $p(i + 1, j)$, does not satisfy the above property $p(i, j) < p(i + 1, j)$, the PAV algorithm replaces both of them by their probability average, that is:

$$p^*(i, j) = p^*(i + 1, j) = \frac{p(i, j) + p(i + 1, j)}{2}, \quad (7)$$

This process is repeated (using the new values) until an isotonic set is reached.

In Table 6, an example of the PAV algorithm, extracted from (Fawcett & Niculescu-Mizil, 2007) is shown. There are 15 instances, 6 negatives and 9 positives. The PAV algorithm begins by sorting the

Table 5.

Bin	Instance	Score
1	e_1 e_2 e_3 e_4	0.95 0.94 0.91 0.90
2	e_5 e_6 e_7 e_8	0.87 0.85 0.80 0.76
3	e_9 e_{10} e_{11} e_{12}	0.70 0.66 0.62 0.62
4	e_{13} e_{14} e_{15} e_{16}	0.51 0.49 0.48 0.48
5	e_{17} e_{18} e_{19} e_{20}	0.45 0.44 0.44 0.42

instances in decreasing order by score and assigning probability estimates of 1 for each positive example and 0 for each negative example. The algorithm iteratively looks for adjacent violators: a local non-monotonicity in the sequence. Initially, adjacent violators (a zero followed by a one) exist at instance pairs 2–3, 6–7, 9–10 and 12–13.

The algorithm operates from the bottom of the instance sequence to the top. First, in step a1, the violation generated by instances 12 and 13 is removed by pooling the two instances together and assigning them a probability estimate of 1/2 (see column a1). This introduces a new violation between instance 11 and the adjacent group 12–13. To remove this new violation, in step a2, instance 11 and the group 12–13 are pooled together, forming a pool of three instances (two negatives and one positive) whose probability estimate is 1/3. The result is shown in column a2.

Next, instances 9–10 (one negative and one positive) are pooled, assigning a probability of 1/2 to each instance. The result is shown in column b.

In steps c1 and c2, the violations between instances 6–8 (one negative and two positives) are removed in two steps. Similarly, instances 2–5 (one negative and three positives) are pooled into a group of probability 3/4 (the intermediate steps are omitted). The final result is shown in column d. The sequence of probability estimates is now monotonically decreasing and no violators remain. This sequence can now be used as the basis for a function that maps classifier scores into probability estimates.

Platt's Method

(Platt, 1999) presents a parametric approach for fitting a sigmoid that maps estimated probabilities into calibrated ones. This method was developed to transform the outputs of a support vector machine (SVM)

Table 6.

#	Score	Probabilities						
		Initial	a1	a2	b	c1	c2	d
0	0.9	1	1	1	1	1	1	1
1	0.8	1	1	1	1	1	1	1
2	0.7	0	0	0	0	0	0	0.75
3	0.6	1	1	1	1	1	1	
4	0.55	1	1	1	1	1	1	
5	0.5	1	1	1	1	1	1	
6	0.45	0	0	0	0	0.5	0.67	0.67
7	0.4	1	1	1	1			
8	0.35	1	1	1	1			
9	0.3	0	0	0	0.5	0.5	0.5	0.5
10	0.27	1	1	1				
11	0.2	0	0	0.33	0.33	0.33	0.33	0.33
12	0.18	0						
13	0.1	1						
14	0.02	0	0	0	0	0	0	0

from the original values $[-\infty, \infty]$ to probabilities, but it can be extended to other types of models or probability variations. The idea consists of applying a sigmoid function to the values of the form:

$$p^*(i, j) = \frac{1}{1 + e^{A \times p(i, j) + B}}, \quad (8)$$

The parameters A and B are determined so that they minimise the negative log-likelihood of the data.

Platt's method is most effective when the distortion in the predicted probabilities has a sigmoid form (as in the above example). Isotonic regression is more flexible and can be applied to any monotonic distortion. Nevertheless, isotonic regression is used to prevent overfitting problems in some cases. Also, all the above methods can use the training set or an additional validation set for calibrating the model. The quality of the calibration may depend on this possibility and the size of the dataset. This is a recurrent issue in calibration, and it has been shown that some methods are better than others for small calibration sets (i.e., Platt's scaling is more effective than isotonic regression when the calibration set is small (Caruana & Niculescu-Mizil, 2004)).

Other Related Calibration Methods

Apart from the methods for obtaining calibrated probabilities, there are other calibration techniques that are only applicable to specific learning methods. For instance, smoothing by m-estimate (Cestnik, 1990) and Laplace (Provost & Domingos, 2000) are other alternative ways of improving the probability estimates given by an unpruned decision tree. Probabilities are generated from decision trees as follows. Let T be a decision tree and let l be a leaf that contains n training instances. If k of these instances are of one class (for instance, of the positive class), then when T is applied to classify new examples, it

assigns a probability of $p = \frac{k}{n}$ that each example i in l belongs to the positive class. However, using the frequencies derived from the count of instances of each class in a leaf might not give reliable probability estimates (for instance, if there are few instances in a leaf). Therefore, for a two-class problem,

the Laplace correction method replaces the probability estimate by $p' = \frac{k+1}{n+2}$. For a more general multiclass problem with C classes, the Laplace correction is calculated as $p' = \frac{k+1}{n+C}$. As Zadrozny

and Elkan (2001) state “the Laplace correction method adjusts probability estimates to be closer to 1/2, which is not reasonable when the two classes are far from equiprobable, as is the case in many real-world applications. In general, one should consider the overall average probability of the positive class should be considered, i.e., the base rate, when smoothing probability estimates” (p. 610). Thus, smoothing by

m-estimate consists of replacing the above-mentioned probability estimate by $p' = \frac{k+b \cdot m}{n+m}$, where b

is the base rate and m is the parameter for controlling the shift towards b . Given a base rate b , Zadrozny and Elkan (2001) suggest using m such that $b \cdot m = 10$.

Another related technique that is also applicable to decision trees is curtailment (Zadrozny & Elkan, 2001). The idea is to replace the score of a small leaf (i.e., a leaf with few training instances) with the

estimate of its parent node, if it contains enough examples. If the parent node still has few examples, the same process is repeated with its parent node and so on until either a node is sufficiently populated or the tree root is reached.

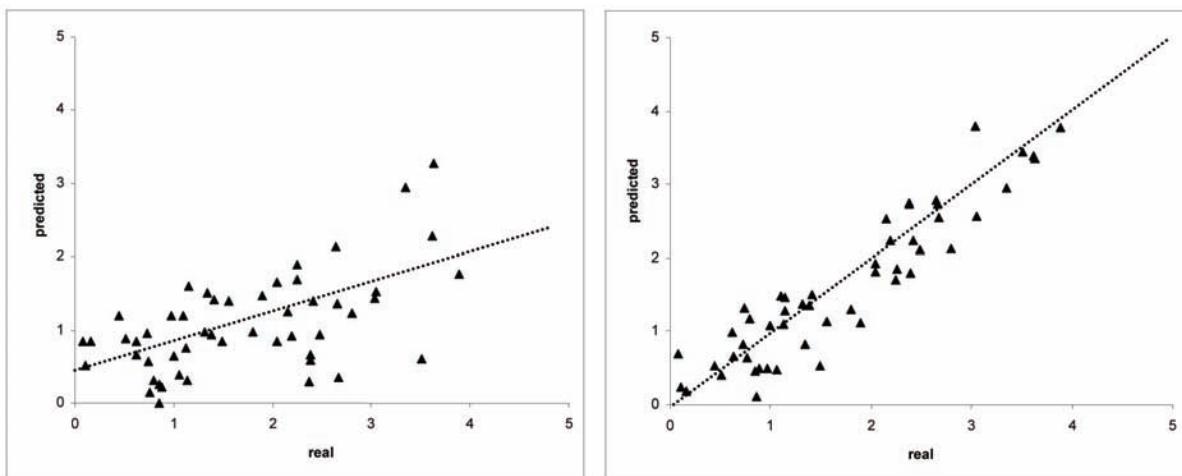
Calibration Methods for RD

Most regression techniques to date have implicitly or explicitly considered the case when the goal is to have the expected output to be equal (or close) to the real average output (type RD). This is because there are two numeric outputs (the predicted value and the real value), so there is greater variety of corrective functions to apply. Figure 2 depicts a comparison of the behaviour of the test data, denoted by “real”, and the model that is to be calibrated (“predicted”).

As stated in the introduction, the characteristic of a calibrated model for type RD is that the errors are equally distributed for the different output values. In other words, the expected value from distinct functions between the predicted value and the real value must be the accurate one according to the function. In one case, for example, the expected value of the difference between the estimated value (y_{est}) and the real value (y) must be near to zero, so $E(y_{est} - y) = 0$. If this value is less than 0, then, on average, the real values are a little higher than the estimated ones; and if this value is greater than 0, on average, the real values are a little lower than the estimated ones. Most regression models usually have this difference quite well calibrated. However, in another case, the expected value of the quotient between the estimated value and the real value should be near to one, so $E(y_{est} / y) = 1$. If this quotient is greater than one, the error is usually positive for high values and negative for low values. Techniques such as linear regression usually give calibrated models, but others (nonlinear regression, local regression, neural networks, decision trees, etc.) can give uncalibrated models.

Logically, in both cases, the errors can be corrected by decreasing the high values and increasing the low values. In general, the solution comes from obtaining some type of estimation of the decalibration function between the real values and the predicted values. One of the most common approximations consists of calculating a linear regression (as in the plots shown in Figure 2) and applying it to the model

Figure 2. Calibration of regression models. Left: an uncalibrated model. Right: a calibrated model.



in order to fit the calibration. These calibrations usually increase the mean squared error $\frac{(y - y_{est})^2}{n}$, but can reduce the relative mean squared error $\frac{(y - y_{est})^2}{(y - \text{mean}(y_{est})) \times n}$ or the error tolerance.

When the decalibration function is nonlinear (but it has a pattern), the problem of calibration becomes more complex, and some kind of nonlinear or local regression is needed to calibrate the model. In these cases, it is not considered to be a calibration process, as such, but rather a meta-learner with several stages (stacking, cascading, etc.).

Calibration Methods for RP

For type RP, the prediction is a probability density function, which is what must be improved. This is a much more complex problem since improving this type of calibration can be done by mangling the prediction estimates (i.e., the MSE can be increased as the result of calibrating). Consequently, a trade-off must be found.

In (Carney & Cunningham, 2006), they approach the problem by formulating it as a multi-objective optimisation problem. In fact, the two objectives of density forecasting are sharpness (a classical quality criterion based on negative log-likelihood (NLL) that tries to maximise the probability density at the observation) and calibration (using the Anderson-Darling (A^2) test over the probability integral transform in order to achieve empirical consistency in the probability estimates). The authors proposed an approach which consists of applying an “a posteriori” multi-objective evolutionary algorithm (MOEA). “A posteriori” means that it simultaneously optimises the two above-mentioned factors by finding a set of non-dominant solutions (called the Pareto-optimal front) from which the user can select the model which best adapts to its objectives.

In this evolutionary approach, the population contains a set of models which are represented by means of a vector of parameters. Hence, “any density forecasting model that can be represented as a vector of values can be optimised using this framework” (Carney, Cunningham, & Lucey, 2006, p.15). Also, at each step of the algorithm, the objective functions (NLL and A^2) are computed in order to determine if the model must be included in the Pareto set. Finally, evolution of the population is achieved by applying a mutation function. The approach can be generalized to use any other objective function as well as any evolutionary algorithm. The authors have applied their approach to two classes of models and MOEA’s: Gaussian Mixture Model (GMM) using a Pareto Mixture Density Network (MDN) and Generalised Autoregressive Conditional Heteroscedasticity (GARCH) models.

FUTURE TRENDS

Future trends in calibration include a clearer recognition of the effects that calibration has and when and how the four types of calibration are related as well as how they relate to classical quality metrics. Calibration and traditional measures are sometimes conflicting, and a couple of metrics (such as A^2 and NLL in the case of type RP), or a hybrid one (such as MSE^p in the case of type CP) should be used to select the best model. A detailed study can be found in Ferri et. al (2008), which shows that the correlation between calibration and traditional measures is not so high.

In classification, most of the calibration methods analysed here work for binary problems. Most calibration methods are based on sorting the instances and/or making bins. However, for more than two classes, it is not so clear how to sort the instances or, more generally, how to make the bins. This is one of the reasons why the multiclass calibration problem has not been studied in more depth. Nevertheless, there are some works like (Zadrozny & Elkan, 2002) where the multiclass calibration problem has been studied, but using approaches for reducing a multiclass problem to a set of binary problems and then finding an approximate solution for that problem.

Another interesting research line consists of studying in depth the relationship between ROC analysis (or its counterpart for regression, REC analysis (Bi & Bennett, 2003)) and calibration. For instance, the use of ROC analysis techniques for repairing concavities (Flach & Wu, 2005) to solve conflicts between the original class ranking and the new estimated probabilities.

Finally, type RP is a future trend on its own, since it is the most complex case which has most recently received attention.

CONCLUSION

In this chapter, the problem of predictive model calibration has been addressed and have been presented the most well-known calibration techniques for classification and regression.

This study shows that there are evaluation measures for classification that are suitable for calibration (such as logloss, MSE, ...). There are also other measures (for instance, accuracy) which are not suitable. Other measures can be used in conjunction with calibration measures, especially the separability measures (AUC). Similarly, for regression, specific measures are needed to evaluate calibration, although they must usually be accompanied by measures of sharpness.

Calibration techniques are usually based on deriving a transformation that converts the values (on types CD and RD) or the probabilities (on types CP and RP) to better estimates. Very different transformation techniques have been devised in the literature, but they usually include some kind of binning or sorting in discrete cases (classification), or some kind of integral in the continuous cases (regression).

REFERENCES

- Ayer, M., Brunk, H., Ewing, G., Reid, W., & Silverman, E. (1955). An empirical distribution function for sampling with incomplete information. *Annals of Mathematical Statistics*, 26(4), 641–647. doi:10.1214/aoms/1177728423
- Bi, J., & Bennet, P. (2003). Regression error characteristic curves. In *Proceedings of the 20th International Conference on Machine Learning* (pp. 43-50).
- Carney, M., & Cunningham, P. (2006). Making good probability estimates for regression. In *Proceedings of the 17th European Conference on Machine Learning* (LNCS 4212, pp. 582-589). Berlin, Germany: Springer.
- Carney, M., Cunningham, P., & Lucey, B. M. (2006). Making density forecasting models statistically consistent. *IIS Discussion Paper Series*.

- Caruana, R., & Niculescu-Mizil, A. (2004). Data mining in metric space: An empirical analysis of supervised learning performance criteria. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 69-78).
- Cestnik, B. (1990). Estimating probabilities: A crucial task in machine learning. In *Proceedings of the 9th European Conference on Artificial Intelligence* (pp. 147-149).
- DeGroot, M., & Fienberg, S. (1982). The comparison and evaluation of forecasters. *The Statistician*, 31(1), 12–22.
- Dowe, D. L., Farr, G. E., Hurst, A. J., & Lentin, K. L. (1996). Information-theoretic football tipping. In *Proceedings of the 3rd Conference on Maths and Computers in Sport* (pp. 233-241).
- Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27, 861–874. doi:10.1016/j.patrec.2005.10.010
- Fawcett, T., & Niculescu-Mizil, A. (2007). PAV and the ROC convex hull. *Machine Learning*, 68(1), 97–106. doi:10.1007/s10994-007-5011-0
- Ferri, C., Hernández-Orallo, J., & Modroiu, R. (2008). An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1), 27–38. doi:10.1016/j.patrec.2008.08.010
- Flach, P. A., & Matsubara, E. T. (2007). A simple lexicographic ranker and probability estimator. In *Proceedings of the 18th European Conference on Machine Learning* (pp. 575-582).
- Flach, P. A., & Wu, S. (2005). Repairing concavities in ROC curves. In *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 702-707).
- Good, I. J. (1952). Rational decisions. *Journal of the Royal Statistical Society. Series B. Methodological*, 14, 107–114.
- Good, I. J. (1968). Corroboration, explanation, evolving probability, simplicity, and a sharpened razor. *The British Journal for the Philosophy of Science*, 19, 123–143. doi:10.1093/bjps/19.2.123
- Lachiche, N., & Flach, P. A. (2003). Improving accuracy and cost of two-class and multi-class probabilistic classifiers using ROC curves. In *Proceedings of the International Conference on Machine Learning* (pp. 416-423).
- Montgomery, D. C., Peck, E. A., & Vining, G. G. (2001). *Introduction to Linear regression analysis*. New York: John Wiley & Sons, Inc.
- Murphy, A. H. (1972). Scalar and vector partitions of the probability score: Part ii. N-state situation. *Journal of Applied Meteorology*, 11, 1182–1192. doi:10.1175/1520-0450(1972)011<0273:SAVPOT>2.0.CO;2
- O'Brien, D. B., Gupta, M. R., & Gray, R. M. (2008). Cost-sensitive multi-class classification from probability estimates. In *Proceedings of the 25th International Conference on Machine Learning* (pp. 712-719).
- Platt, J. C. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in large margin classifiers* (pp. 61-74). Cambridge, MA: MIT Press.

Provost, F., & Domingos, P. (2000). *Well-trained PETs: Improving probability estimation trees* (Tech. Rep. CDER #00-04-IS). Stern School of Business, New York University.

Sanders, F. (1963). On subjective probability forecasting. *Journal of Applied Meteorology*, 2, 191–201. doi:10.1175/1520-0450(1963)002<0191:OSPF>2.0.CO;2

Zadrozny, B., & Elkan, C. (2001). Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In *Proceedings of the 8th International Conference on Machine Learning* (pp. 609-616).

Zadrozny, B., & Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 694-699).

KEY TERMS AND DEFINITIONS

Calibration Technique: any technique that aims to improve probability estimation or to improve error distribution of a given model.

Distribution Calibration in Classification (or simply “class calibration”): the degree of approximation of the true or empirical class distribution with the estimated class distribution.

Probabilistic Calibration for Classification: any technique that improves the degree of approximation of the predicted probabilities to the actual probabilities.

Distribution Calibration in Regression: any technique that reduces the bias on the relation between the expected value of the estimated value and the mean of the real value.

Probabilistic Calibration for Regression: for “density forecasting” models, in general, any calibration technique that makes these density functions be specific for each prediction, narrow when the prediction is confident, and broader when it is less so

Calibration Measure: any kind of quality function that is able to assess the degree of calibration of a predictive model.

Confusion Matrix: a visual way of showing the recount of cases of the predicted classes and their actual values. Each column of the matrix represents the instances in a predicted class, while each row represents the instances in an actual class

Reliability Diagrams: In these diagrams, the prediction space is discretised into 10 intervals (from 0 to 0.1, from 0.1 to 0.2, etc.). The examples whose probability is between 0 and 0.1 go into the first interval, the examples between 0.1 and 0.2 go into the second, etc. For each interval, the mean predicted value (in other words, the mean predicted probability) is plotted (x axis) against the fraction of positive real cases (y axis). If the model is calibrated, the points will be close to the diagonal

ENDNOTE

¹ Scores can also be used (Zadrozny & Elkan, 2002)

Chapter 7

Classification with Incomplete Data

Pedro J. García-Laencina

Universidad Politécnica de Cartagena, Spain

Juan Morales-Sánchez

Universidad Politécnica de Cartagena, Spain

Rafael Verdú-Monedero

Universidad Politécnica de Cartagena, Spain

Jorge Larrey-Ruiz

Universidad Politécnica de Cartagena, Spain

José-Luis Sancho-Gómez

Universidad Politécnica de Cartagena, Spain

Aníbal R. Figueiras-Vidal

Universidad Carlos III de Madrid, Spain

ABSTRACT

Many real-word classification scenarios suffer a common drawback: missing, or incomplete, data. The ability of missing data handling has become a fundamental requirement for pattern classification because the absence of certain values for relevant data attributes can seriously affect the accuracy of classification results. This chapter focuses on incomplete pattern classification. The research works on this topic currently grows wider and it is well known how useful and efficient are most of the solutions based on machine learning. This chapter analyzes the most popular and proper missing data techniques based on machine learning for solving pattern classification tasks, trying to highlight their advantages and disadvantages.

DOI: 10.4018/978-1-60566-766-9.ch007

INTRODUCTION

Pattern classification is the discipline of building machines to classify data (patterns or input vectors) based on either a priori knowledge or on statistical information extracted from the patterns (Bishop, 1995; Duda *et al.*, 2000; Jain *et al.*, 2000; Ripley, 1996). This research field was developed starting from the 1960's, and it has progressed to a great extend in parallel with the growth of research on knowledge-based systems and artificial neural networks. Pattern classification has been successfully applied in several scientific areas, such as computer science, engineering, statistics, biology, and medicine, among others. These applications include biometrics (personal identification based on several physical attributes as fingerprints and iris), medical diagnosis (CAD, computer aided diagnosis), financial index prediction, and industrial automation (fault detection in industrial process). Many of these real-word applications suffer a common drawback, missing or unknown data (incomplete feature vector). For example, in an industrial experiment some results can be missing because of mechanical/electronic failures during the data acquisition process (Lakshminarayan *et al.*, 2004; Nguyen *et al.*, 2003). In medical diagnosis some tests are not possible to be done because both the hospital lacks the necessary medical equipment or some medical tests may not be appropriate for certain patients (Jerez *et al.*, 2006; Liu *et al.*, 2005; Markey & Patel, 2004; Proschan *et al.*, 2001). In the same context, another example could be an examination by a doctor, who performs several different kinds of tests; some test results may be available instantly, and some may take several days to complete. Anyway, it might be necessary to reach a preliminary diagnosis instantly, using only test results that are available. Missing data is a subject which has been treated extensively in the literature of statistical analysis (Allison, 2001; Little & Rubin, 2002; Schaffer, 1997), and also, but with less effort, in the pattern recognition literature. The unavailability of the data hinders the decision making processes due to the dependencies of decisions on information. Most scientific, business and economic decisions are somehow related to the information available at the time of making such decisions. As an example, most business evaluations and decisions are highly dependent on the availability of sales and other information, whereas advances in research are based on discovery of knowledge from various experiments and measured parameters. The ability of handling missing data has become a fundamental requirement for pattern classification because inappropriate treatment of missing data may cause large errors or false results on classification. In addition, it is being a more common problem in real-world data. Another clear example of the importance of handling missing data is that 45% of data sets in the UCI repository have missing values, what is one of most used collection of data sets for benchmarking machine learning procedures.

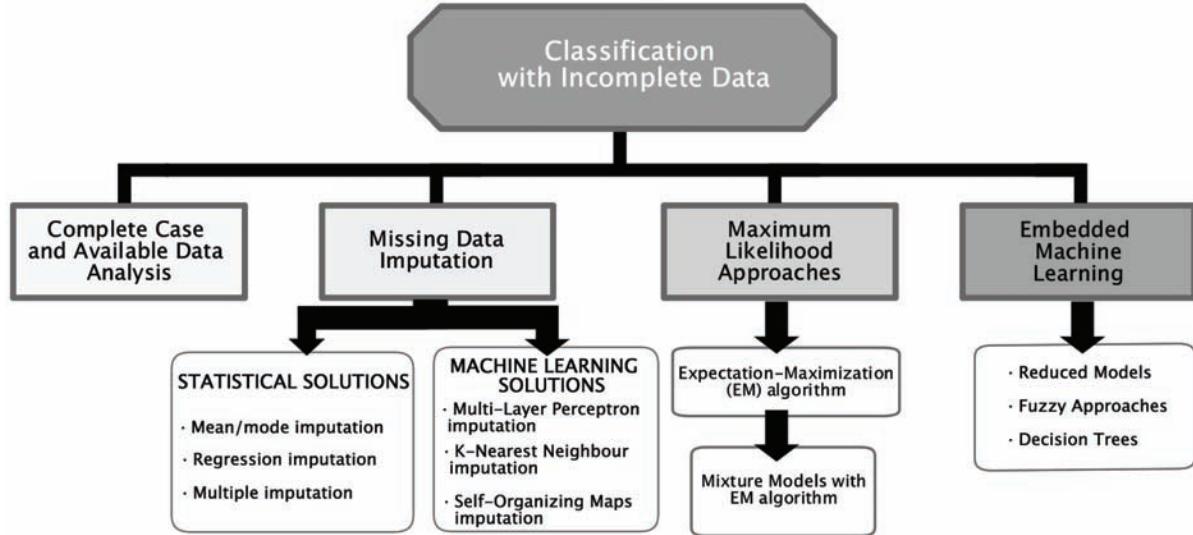
In general, pattern classification with missing data concerns two different problems, handling missing values and pattern classification. Most of the approaches in the literature can be grouped in four different types of approaches depending on how both problems are solved. Figure 1 resumes the different approaches in pattern classification with missing data.

Intuitively the easiest way to deal with missing values is simply deleting the incomplete data. In a multivariate environment missing values may occur on one or more attributes and missing components are often a significant portion of the whole data set, and so, the deletion of these incomplete items may cause a substantial loss of information.

Another approach to handle missing data is to try to estimate the missing data. The process of estimating the missing data components is referred to as imputation. This chapter distinguishes between two different types of imputation methods,

Classification with Incomplete Data

Figure 1. Methods for pattern classification with incomplete data. This scheme shows the different procedures that are analyzed in this chapter.



- Statistical imputation procedures, such as mean or multiple imputation.
- Imputation based on machine learning approaches, such as Multi-Layer Perceptron (MLP), *K*-Nearest Neighbours (KNN), or Self-Organizing Maps (SOM).

Other broad type of methods is the model-based procedures. In this approach, a model is defined for the variables with missing values and making statistical inferences based on Maximum Likelihood (ML) methods. There are several ways to obtain ML estimators, and one of the most common is the Expectation-Maximization (EM) algorithm. Finally, we consider a last group of machine learning methods which incorporates handling missing data into the learning process of the classification task. In particular, reduced methods, decision trees and fuzzy neural approaches are described. These methods work very well in many situations, but they do not estimate missing values, which is a necessary characteristic in some applications.

This chapter reviews the most important missing data techniques in pattern classification, trying to highlight their advantages and disadvantages. The remainder of this work is structured as follows. The next section introduces the notation for incomplete data classification, and after that, we describe some real incomplete data applications and the different missing data mechanisms. In the rest of this chapter, several methods for dealing with missing data are discussed. Our aim is to show the most proper and useful procedures for handling missing data, doing a special effort with solutions based on machine learning.

INCOMPLETE DATA CLASSIFICATION

In pattern classification (Duda *et al.*, 2000), a pattern¹ is represented by a vector of d features or attributes (continuous or discrete), i.e., $\mathbf{x}=[x_1, x_2, \dots, x_i, \dots, x_d]$. For example, a pattern could be: an audio signal, being the corresponding feature vector its frequency spectral components, or a digitized image for a character recognition problem. In addition, each pattern belongs to one of c possible classes. In general, a decision making tool operates in two modes: training (learning) and classification (testing). During the training phase, the classifier is completely designed (to set up its internal parameters) using a set of training samples, which is named training set. We consider that training samples are labeled (supervised learning), i.e., the label t of a training pattern \mathbf{x} represents the class to which \mathbf{x} belongs. In the testing mode, the trained system assigns the input pattern to one of the classes under consideration based on its attributes. The aim of designing a decision making tool is to classify future test samples, which were not used during the training stage. Datasets are often characterized by incompleteness. It is unfortunate that for some reasons, input data are not obtainable for every pattern of every attribute. A missing value is denoted by ‘?’ symbol; thus, the n -th pattern $\mathbf{x}_n=[?, -0.2, ?, 0.3]$ presents missing values at first and third attributes. In our notation, the binary vector \mathbf{m} indicate which features of \mathbf{x} are unknown, such that $m_i=1$ if the feature x_i is missing, and $m_i=0$ if the input feature is present. In the previous example, $\mathbf{m}_n=[1, 0, 1, 0]$. In order to resume the notation, let us to assume a data set D composed of N labeled patterns,

$$D = \{\mathbf{X}, \mathbf{T}, \mathbf{M}\} = \left\{ (\mathbf{x}_n, t_n, \mathbf{m}_n) : \mathbf{x}_n = \left\{ x_{i,n} \right\}_{i=1}^d, x_{i,n} \text{ is missing} \Leftrightarrow m_{i,n} = 1 \right\}_{n=1}^N \quad (1)$$

where \mathbf{x}_n is the n -th input vector composed of d features; labelled as t_n , with C possible classes; and \mathbf{m}_n are binary variables that indicates which input features are unknown in \mathbf{x}_n . We assume that the dataset \mathbf{X} can be divided into an observed component \mathbf{X}^o and a missing component \mathbf{X}^m , and each input vector may have different combinations of missing features.

Missing Data

As we have already mentioned in the introduction, missing data is a common drawback that appears in many real-world situations. In surveys (Rubin, 1987; Wang & Fan, 2004), it is virtually assured that a certain level of non-response will occur, e.g., partial non-response in a polls or entered data is partially erroneous. In control-based applications (Lakshminarayan *et al.*, 2004; Ji & Elwaid, 2000; Nguyen *et al.*, 2003), missing values appears due to failures of monitoring or data collector equipment, disruption of communication between data collectors and the central management system, failure during the archiving system (hardware or software), etc. Wireless sensor networks (Halatchev & Gruenwald, 2005; Mohammed *et al.*, 2006) also suffers unknown data due to different reasons, as power outage at the sensor node, random occurrences of local interferences or a higher bit error rate of the wireless radio transmissions. In automatic speech recognition (Cooke *et al.*, 1994; Parveen *et al.*, 2004), speech samples what are corrupted by very high levels of noise are also treated as missing data. Another example is incomplete observations in financial and business applications (DiCesare, 2006; Sharpe & Kofman, 2003 like credit assignment (unknown information about the credit petitioner) or financial-time series forecasting (there is no stock price data available for regular holidays). A recently application in which incomplete data ap-

pears is biology research with DNA microarrays (Troyanskaya *et al.*, 2001; Kim *et al.*, 2004), where the gene data may be missing due to various reasons such as scratch on the slide or contaminated samples. Missing values are also common in medical diagnosis, e.g., a medical practitioner may not order a test whose outcome appears certain or not relevant to the diagnosis, or a feature can be missing because it proved to be difficult/harmful to measure (Jerez *et al.*, 2006; Liu *et al.*, 2005; Markey & Patel, 2004; Proschan *et al.*, 2001).

Missing Data Mechanisms

The appropriate way to handle incomplete input data depends in most cases upon how data attributes became missing. Little and Rubin (2002) define several unique types of missing data mechanisms.

- *Missing Completely At Random* (MCAR). MCAR situation occurs when the probability that a variable is missing is independent of the variable itself and any other external influence. The available variables contain all the information to make the inferences. The reason for missingness is completely at random, i.e., the probability that an attribute is missing is not related to any other features. As an example, suppose weight and age are variables of interest for a particular study. If the likelihood that a person will provide his or her weight information is the same for all individuals regardless of their weight or age, then the missing data is considered to be MCAR.
- *Missing At Random* (MAR). The missingness is independent of the missing variables but the pattern of data missingness is traceable or predictable from other variables in the database. An example is a sensor that fails occasionally during the data acquisition process due to power outage. In this example, the actual variables where data are missing are not the cause of the incomplete data. Instead, the cause of the missing data is due to some other external influence.
- *Not Missing At Random* (NMAR). The pattern of data missingness is non-random and depends on the missing variable. In this situation, the missing variable in the NMAR case cannot be predicted only from the available variables in the database. If a sensor cannot acquire information outside a certain range, its data are missing due to NMAR factors. Then the data is said to be censored. If missing data are NMAR, valuable information is lost from the data; and there is a no general method of handling missing data properly.

When data are MCAR or MAR, the missing data mechanism is termed ignorable. Ignorable mechanisms are important because when they occur, a researcher can ignore the reasons for missing values in the analysis of the dataset, and thus simplify the model-based methods used for missing data analysis. For this reason, majority of research covers the cases where missing data is of the MAR or the MCAR type. In the next sections, several approaches to deal with missing values are explained.

When discussing the application of machine learning algorithms to data sets including missing values, two scenarios can be distinguished:

- It may be that the data used for training is complete, and values are missing only in the test data.
- Values may be missing both in training and test data.

The first case is not a common situation, because the classifier has been designed using complete cases, and any assumption about missing data has not been done during its training. In this situation,

there are two possible solutions, incomplete cases have to be excluded from the data set, or missing values have been filled in by estimations (imputed values). This preprocessing is often necessary to enable the training of models, e.g. when using standard Multi-Layer Perceptrons (MLPs), which cannot deal with missing values. The second itemized case is more natural, when training and test data are gathered and processed in a highly similar way, and so, the classifier is trained considering that input vectors may be incomplete.

DELETION OF INCOMPLETE DATA

When faced with missing values, the *complete-case* and *available-case* analyses are common, but not recommended, procedures used in order to force the incomplete data set into a rectangular complete-data format (Little & Rubin, 2002; Schaffer, 1997). The first approach is also known as *listwise* or *casewise deletion*, i.e., omit the cases that have missing data for any one variable used in a particular analysis. This procedure can be justified whenever the large quantity of data is available (in general, only 5% of missing data is an acceptable amount to delete from the data set). Its advantage is the possibility of using directly the standard pattern classification methods for complete data, but a test pattern with missing values cannot be classified because the deletion process will omit it. The second approach is the *available-case analysis*, in which only the cases with available variables of interest are used. In general, available-case analysis makes better use of the data than complete case analysis (this method uses all the available values); but its disadvantage is that the number of attributes is not the same for all patterns (it depends on what features are incomplete for each pattern), and so, the standard classification methods cannot be directly applied. These approaches are simple, but the major drawback of both procedures is the loss of information that occurs in the deletion process. As an example, consider a dataset with 25 input features and each feature has a 5% probability of being missing, then, there will be a less than 30% probability that there is a complete pattern with its 25 variables. To be precise, if this dataset was to be observed according to Bernoulli process, only $0.95^{25} = 0.276$ is the expected proportion of input vectors to be complete. It will essentially imply that if there are 100 instances of recorded data and casewise deletion is applied, only 28 instances are likely to be used.

MISSING DATA IMPUTATION METHODS

Most decision making tools such as the commonly used MLPs, Support Vector Machines (SVMs) and many other machine learning techniques can not be used for decision making if data are not complete. In cases of incomplete data vectors, the first step toward decision making is to estimate the missing values. This pre-processing step is known as *imputation*. Imputation is a procedure for entering a value for a specific data item where the feature is missing or unusable.

Consider an input data set

$$\mathbf{X} = (\mathbf{X}^o, \mathbf{X}^m) \quad (2)$$

which can be divided into its observed component \mathbf{X}^o (complete patterns) and its missing component \mathbf{X}^m (incomplete patterns, i.e., input vectors with missing features). The main objective of an imputation procedure is to implement an accurate missing data estimator $f(\cdot)$, which is trained by means of \mathbf{X}^o . The model $f(\cdot)$ is defined by a set of parameters, and its optimal values are obtained during the training stage. After that, missing feature values contained into \mathbf{X}^m are estimated using $f(\cdot)$. In our notation,

$$\tilde{\mathbf{X}} = (\mathbf{X}^o, \tilde{\mathbf{X}}^m) \quad (3)$$

denotes the imputed dataset, which is composed of the observed component \mathbf{X}^o and the imputed component $\tilde{\mathbf{X}}^m$, i.e., incomplete patterns whose missing values have been filled in using the estimated values from $f(\mathbf{X}^m)$.

Once missing values have been estimated, pattern recognition tools for decision making can then be used. The challenge missing data pose to the decision making process is more evident in on-line applications where data has to be used almost instantly after being obtained. In a case where some variables are not measured, it becomes difficult to continue with the decision making process. The biggest challenge is that the standard machine learning techniques are not able to process input data with missing values and hence, cannot perform classification tasks. The main advantage of imputation methods is that they can be used in conjunction with any complete data classifier. Following the main groups of imputation techniques are outlined: statistical and machine learning based procedures. Finally, a brief analysis of the performance measures on imputation methods ends this section.

Statistical Based Techniques for Imputation

The concept of imputation has been studied extensively in the literature of statistical analysis (Little & Rubin, 2002; Schaffer, 1997). In this research area, the main imputation procedures are mean/mode, regression-based and multiple imputation solutions. All of them are described next.

Figure 2. Mean/mode imputation method in a simple four dimensional data set. Imputed values are shown in bold face.

	x_1	x_2	x_3	x_4
\mathbf{x}_1	-0.20	0	?	1
\mathbf{x}_2	0.10	?	Red	0
\mathbf{x}_3	?	1	Green	1
\mathbf{x}_4	0.21	0	Green	1
\mathbf{x}_5	?	1	Red	?
\mathbf{x}_6	-0.10	0	Blue	?
\mathbf{x}_7	0.20	?	Red	0

Mean/Mode
Imputation

	x_1	x_2	x_3	x_4
\mathbf{x}_1	-0.20	0	Red	1
\mathbf{x}_2	0.10	0	Red	0
\mathbf{x}_3	0.03	1	Green	1
\mathbf{x}_4	0.21	0	Green	1
\mathbf{x}_5	0.03	1	Red	1
\mathbf{x}_6	-0.10	0	Blue	1
\mathbf{x}_7	0.20	0	Red	0

Mean and Mode Imputation

The earliest used method of imputation was unconditional mean imputation. In this approach, missing components of a vector are filled in by the average value of that component in all the observed cases. Another possibility is class-conditional mean imputation, where missing data of an input pattern are estimated by the average from the complete cases that belongs to the same class as the incomplete pattern. When the incomplete attribute is discrete, the most frequent value (mode) for this attribute in the whole dataset is used to fill up missing values. In order to show this procedure, consider a four dimensional input data set, which is composed of continuous and discrete (binary and categorical) attributes. In this example, all attributes are incomplete, and the missing values are estimated by the mean/mode imputation method. As it can be observed in Figure 2, given an incomplete feature, all its missing components are replaced by the same imputed value, i.e., mean or mode of the attribute of interest, without considering the known values of the remaining complete features. Imputed values are shown in bold face.

The mean/mode imputation method has the obvious disadvantages that it under represents the variability in the data, and it also completely ignores the correlations between the various components of the data.

Regression Imputation

Regression imputation is well suited when the missing variables of interest are correlated with the data that is available in the complete sample. The missing components are filled in by the predicted values from a regression analysis using the components of the vector that are present. The method of regression to be used depends on the nature of the data (Little & Rubin, 2002; Schaffer, 1997). Linear regression can be used if the variable dependency follows a linear relationship. On the other hand in non-linear regression the purpose is to fit a curve to the data and to find the required points from the curve. When regression is used to impute missing values on independent variables, this will contribute to multicollinearity because the imputed values for missing data will be perfectly correlated with the rest of the variables in the model. The disadvantage of this approach is that all the imputed values follow a single regression curve and cannot represent any inherent variation in the data.

Multiple Imputation

The two described approaches above provide a simple missing data imputation, which does not reflect the uncertainty about the prediction of the unknown values. Instead of filling in a single value for each missing one, a multiple imputation procedure replaces each missing value with a set of plausible ones that represent the uncertainty about the right value to impute (Little & Rubin, 2002; Rubin, 1987; Schaffer, 1997). The missing values are imputed M times to produce M complete data sets using an appropriate model that incorporates random variation.

The desired analysis is performed on each data set using the standard complete data methods and then the average of parameter estimates across M samples is taken to produce a single point estimate. Standard errors are calculated as a function of average squared standard errors of the M estimates and the variance of the M parameter estimates across samples (Schaffer, 1997). Thus, multiple imputation procedure involves three distinct phases:

1. The missing data is filled in M times to generate M complete data sets.
2. The M complete data sets are analyzed by using standard procedures.
3. The results from the M complete data sets are combined for the inference.

Little and Rubin (2002) conclude that casewise and mean substitution methods are inferior when compared with multiple imputation. Regression methods are somewhat better, but not as good as multiple imputation or other maximum-likelihood approaches. Its main drawback is that learning several imputation models can be a costly operation, and in addition to this, using multiple imputations leads to maintaining an ensemble of classifiers at test time. Combining multiple imputation method with cross validation requires training and evaluating many individual classifiers.

Machine Learning Techniques for Imputation

Imputation methods based on machine learning consist of creating a predictive model to estimate values that will substitute those missing. These approaches model the missing data estimation based on information available in the dataset. There are several options varying from imputation with K Nearest Neighbour (KNN) to imputation procedures based on Self Organizing Maps (SOM). After imputed data fill in the incomplete feature values, it is necessary to train a classifier using the imputed training set.

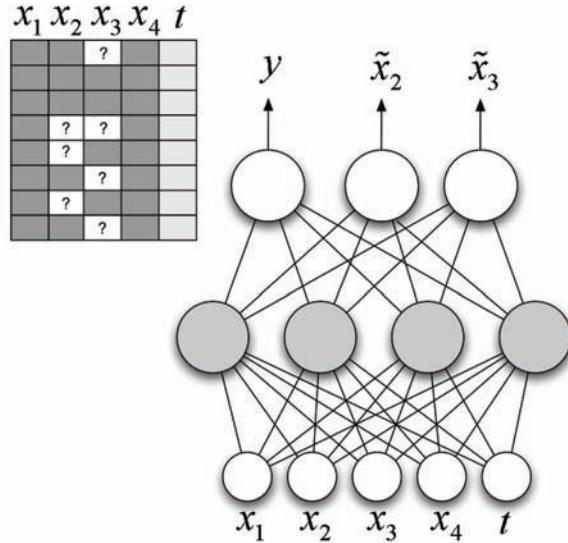
Multi-Layer Perceptron Imputation

The Multi-Layer Perceptron (MLP) is one of the most widely applied and researched artificial neural network models. MLP networks are normally applied to performing supervised learning tasks, which involve iterative training methods to adjust the connection weights within the network. This is commonly formulated as a multivariate non-linear optimization problem over a very high-dimensional space of possible weight configurations. For a complete description of the MLP properties, training procedures, and applications, see (Duda *et al.*, 2000; Bishop, 1995).

MLP networks can be used to estimate missing values by training an MLP to learn the incomplete features (they are used as outputs) given the remaining complete features as inputs (Gupta & Lam, 1996; Nordbotten, 1996; Sharpe & Solly, 1995). This method uses an MLP with a single layer of hidden units as imputation model, and it can be considered as an enhanced version of the regression-based imputation approach. In order to train the network, the Back-Propagation (BP) approach is utilized on the cases without missing items. The training is evaluated according to the Mean Square Error (MSE) of the differences between the predicted-values and observed values. Then, the optimal weights of BP can be determined at which the MSE reaches the convergence according to a pre-established criteria. The MLP imputation scheme can be described as follows:

1. Given an incomplete input dataset \mathbf{X} , separate the input vectors that do not contain any missing data (observed component, \mathbf{X}^o) with the ones that have missing values (missing component, \mathbf{X}^m).
2. For each possible combination of incomplete attributes in \mathbf{X}^m , construct an MLP scheme using \mathbf{X}^o . The target variables are the attributes with missing data, and the input variables are the other remaining attributes (Sharpe & Solly, 1995). In this approach, there is one MLP model per missing variables combination. Depending on the nature of the attributes to be imputed (numerical or

Figure 3. MTL neural network for solving a classification problem of input vectors with four attributes, where the features x_2 and x_3 present incomplete data. This network learns the classification task, and each imputation task associated to an incomplete input feature at the same time. By doing this, missing data imputation is oriented to solve the main classification task.



discrete), different error functions (sum of squares error or cross-entropy error) are minimized during the training process.

3. After the optimal MLP architectures are chosen, for each incomplete pattern in \mathbf{X}^m , unknown values are predicted using its corresponding MLP model (according to the attributes to be imputed).

As an example, consider an input dataset composed of four attributes: x_1, x_2, x_3 and x_4 , where x_2 and x_3 are incomplete, i.e., they present unknown values. In this problem, missing values occurs in all possible combinations of missing features: $\{x_2\}$, $\{x_3\}$ and $\{x_2, x_3\}$; and for each combination of missing attributes, the missing data indicator vectors are respectively: $[0,1,0,0]$, $[0,0,1,0]$ and $[0,1,1,0]$. In this situation, three different MLPs, $f(\cdot)$, $g(\cdot)$ and $h(\cdot)$, have to be designed to impute the missing values in the different combinations of incomplete features:

$$\mathbf{m} = [0, 1, 0, 0] \Rightarrow \tilde{x}_2 = f(x_1, x_3, x_4) \quad (4)$$

$$\mathbf{m} = [0, 0, 1, 0] \Rightarrow \tilde{x}_3 = g(x_1, x_2, x_4) \quad (5)$$

$$\mathbf{m} = [0, 1, 1, 0] \Rightarrow \begin{cases} \tilde{x}_2 = h_{x_2}(x_1, x_4) \\ \tilde{x}_3 = h_{x_3}(x_1, x_4) \end{cases} \quad (6)$$

There is an MLP for each combination of incomplete attributes, and in each one of them, the remaining complete attributes are used as inputs. For example, in the MLP scheme associated to $\mathbf{m}=[0,1,1,0]$,

there are two different outputs, (\tilde{x}_2 and \tilde{x}_3), one for each incomplete attribute, and two inputs (x_1 and x_4), i.e., the observed feature values. All the MLP schemes are trained using the complete patterns.

Due to weights initialization is critical in the MLP training, the previous MLP imputation process is repeated several times in order to obtain a realistic missing data estimation. Thus, several different MLP models are generated per missing variables combination. Imputed values are obtained by averaging the missing data estimation provided by each MLP model. The MLP approach can be an useful tool for reconstructing missing values (Sharpe & Solly, 1995). However, its main disadvantage is that when missing items appears in several combinations of attributes in a high-dimensional problem, a huge number of MLP models have to be designed.

Other MLP methods have been proposed to impute missing values using a different approach. Yoon and Lee (1999) propose the TEST (Training-EStimation-Training) algorithm as a way of using MLP to impute missing data. It consists of three steps. First, the network is trained with all the complete patterns. Secondly, the parameters (weights) are used to estimate the missing data in all incomplete patterns by means of back-propagation in the inputs. Third, the MLP network is trained again using the whole data set, i.e, complete and imputed patterns. However, this procedure cannot estimate missing values in the test set.

In recent years, an MLP imputation procedure has been proposed using the advantages of Multi-Task Learning (MTL) (Caruana, 1997; García-Laencina *et al.*, 2005; García-Laencina *et al.* 2007, Sancho-Gómez *et al.*, 2008; Silver, 2000). MTL is an approach to machine learning that solves a problem (main task) together with other related problems (extra tasks) at the same time, using a shared representation. García-Laencina *et al.* (2007) propose an MTL neural network scheme that combines missing data imputation and pattern classification is proposed. This procedure utilizes the incomplete features as secondary tasks, and learns them in parallel with the main classification task. Figure 3 shows a full-connected neural network based on MTL for solving a decision problem that presents missing values in x_2 and x_3 .

This neural scheme learns three different tasks: one classification task, associated to the network output y , and two imputation tasks, associated to the outputs \tilde{x}_2 and \tilde{x}_3 . In the input layer, there is an input unit for each feature, and also an extra input unit associated to the classification target. This extra input (classification target) is used as hint to learn the secondary imputation tasks. Hidden neurons used in this approach do not work in the same way as standard neurons, because they compute different outputs for the different tasks to be learned (García-Laencina *et al.*, 2008). They do not include the input signal they have to learn in its corresponding output unit in the sum product. For example, in Figure 3, the imputation output \tilde{x}_2 is learned using the information from the attributes x_1 , x_3 and x_4 , and the classification target t , but it does not depend on x_2 . The imputation outputs are used to estimate the missing values during the training stage. By doing this, missing data estimation is oriented to solve the classification task, because the learning of the classification task affects the learning of the secondary imputation tasks.

K-Nearest Neighbors Imputation

The K Nearest Neighbours (KNN) algorithm is part of a family of learning methods known as instance-based (Batista & Monard, 2002; Batista & Monard, 2003; Troyanskaya *et al.*, 2001). Here, we study the performance of the KNN algorithm to impute the missing values. Rather than using all available instances in the data, the KNN imputation algorithm uses only similar cases with the incomplete pattern.

Given an incomplete pattern \mathbf{x} , this method selects the K closest cases from the training cases with known values in the attributes to be imputed (i.e., features with missing values in \mathbf{x}), such that they minimize some distance measure (Batista & Monard, 2002). In our notation, $V = \{\mathbf{v}_k\}_{k=1}^K$ represents the set of K nearest neighbours of \mathbf{x} arranged in increasing order of its distance. Although the K nearest neighbours can also be selected on the instances without any missing value, it is recommended the previous option (Troyanskaya *et al.*, 2001). The optimal value of K is usually chosen by cross-validation. Once the K nearest neighbours have been found, a replacement value to substitute for the missing attribute value must be estimated. How the replacement value is calculated depends on the type of data; for example the mode is frequently selected for discrete data, while the mean is used for numerical data.

Distance Metric

To apply the KNN approach to impute missing data, one of the most important issues is to select an appropriate distance metric, such as the Heterogeneous Euclidean-Overlap Metric (HEOM). This metric computes different attribute distance measures on different types of attributes. The HEOM distance between two input vectors \mathbf{x}_a and \mathbf{x}_b is given by,

$$d(\mathbf{x}_a, \mathbf{x}_b) = \sqrt{\sum_{i=1}^n d_i(x_{i,a}, x_{i,b})^2} \quad (7)$$

where $d_i(x_{i,a}, x_{i,b})$ is the distance between x_a and x_b on its i -th attribute:

$$d_i(x_{i,a}, x_{i,b}) = \begin{cases} 1, & (1 - m_{i,a})(1 - m_{i,b}) = 0, \\ d_O(x_{i,a}, x_{i,b}), & x_i \text{ is a discrete attribute,} \\ d_N(x_{i,a}, x_{i,b}), & x_i \text{ is a numerical attribute.} \end{cases} \quad (8)$$

Unknown data are handled by returning a distance value of 1 (i.e., maximal distance) if either of the input values is unknown. The overlap distance function d_O assigns a value of 0 if the discrete attributes are the same, otherwise a distance value of 1. The range normalized difference distance function d_N is

$$d_N(x_{i,a}, x_{i,b}) = \frac{|x_{i,a} - x_{i,b}|}{\max(x_i) - \min(x_i)} \quad (9)$$

where $\max(x_i)$ and $\min(x_i)$ are respectively the maximum and minimum values observed in the training set for the numerical attribute x_i .

Missing Data Estimation with KNN

Consider that \mathbf{x} presents a missing value on its i -th input feature (i.e., $m_i = 1$). Once its K nearest neighbours have been chosen, $V = \{\mathbf{v}_k\}_{k=1}^K$, the unknown value is estimated using the corresponding i -th attribute values of V .

When the i -th input feature is a discrete variable, the most popular choice is to impute to the mode of the input feature values of its corresponding neighbours. Meanwhile, if the i -th input feature is a continuous variable, different missing data estimation procedures can be done in the KNN approach:

- *Mean estimation.* Imputed value (\tilde{x}_i) is obtained by the mean value of its K nearest neighbours, i.e.,

$$\tilde{x}_i = \frac{1}{K} \sum_{k=1}^K v_{i,k}. \quad (10)$$

- *Weighted mean estimation.* One obvious refinement is to weight the contribution of each \mathbf{v}_k according to their distance to \mathbf{x} , giving greater weight to closer neighbours,

$$\tilde{x}_i = \frac{1}{K} \sum_{k=1}^K w_k v_{i,k}, \quad (11)$$

where w_k denotes the weight associated to the k -th neighbour. A proper choice for w_k is the inverse square of the distance $d(\mathbf{x}, \mathbf{v}_k)$.

As we have already mentioned in the MTL approaches for imputation, a desirable characteristic for an imputation method is that the missing data estimation is aimed at improving the classification accuracy results, i.e., it provides an imputed dataset which improves the classification stage. Following this idea, an efficient KNN imputation procedure using a feature-weighted distance metric has been recently proposed (García-Laencina *et al.*, 2008). This distance metric considers the input attribute relevance for classification according to the Mutual Information (MI) concept (Kwak & Choi, 2002), which has been used as a relevance measure in several feature selection algorithms. This procedure assigns a weight λ_i to each i -th input feature according to the amount of information that this attribute contains about the target class variable. In particular, the feature-weighted distance metric between two input vectors \mathbf{x}_a and \mathbf{x}_b is computed by,

$$d_\lambda(\mathbf{x}_a, \mathbf{x}_b) = \sqrt{\sum_{i=1}^n \lambda_i d_i(x_{i,a}, x_{i,b})^2} \quad (12)$$

where λ_i represents the normalized MI measured between the i -th input feature and the target class variable, and $d_i(x_{i,a}, x_{i,b})$ is the distance metric defined in (8). For each incomplete pattern, its selected K neighbours are used to provide imputed values which can make the classifier design easier, and thus, the classification accuracy is increased. In order to illustrate how this method works, consider a separable binary classification task which consists of four clusters drawn in a two dimensional space. Two clusters belong to the class “circle”, and they are centered on $[0, 0]$ and $[-0.5, -0.2]$. The remaining two clusters are labeled with the class “square”, being centered on $[-0.4, -0.6]$ and $[-0.2, +0.4]$. Now, twelve completely irrelevant attributes are added to the original dataset. They are random variables which are uniformly distributed between -1 and 1 . For these irrelevant variables, the MI between each one of them

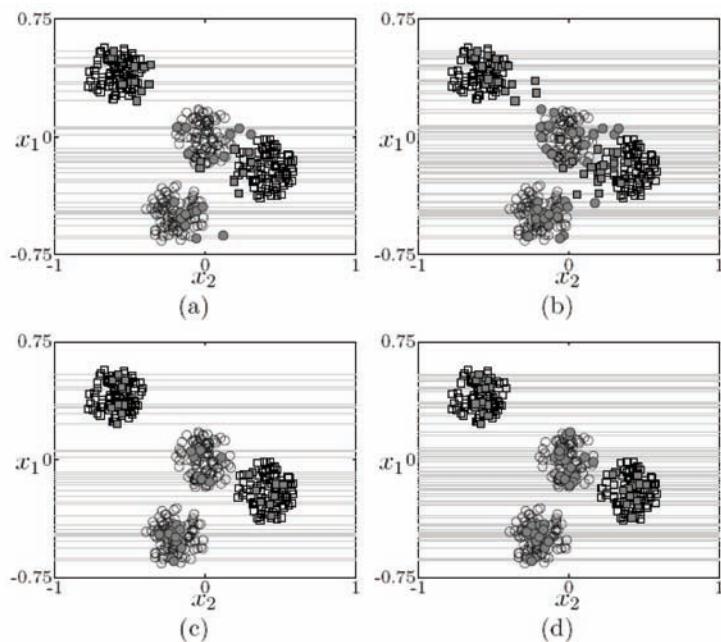
and the classification task is theoretically equal to zero. These twelve random variables are added in order to evaluate the irrelevant attributes influence on the imputation stage. For doing this, 10% and 20% of missing data are randomly inserted into the second feature. Figure 4 shows the imputed datasets by the standard KNN method and the KNN approach based on MI, by considering five neighbours ($K=5$). Horizontal lines denote incomplete input vectors with unknown data in x_2 . Depending on its class, imputed patterns are “squares” or “circles” filled in gray color. In Figure 4(a), we can observe that the imputed values provided by the standard KNN method distorts the input data distribution due to the presence of irrelevant attributes. After imputation is done by KNN, the obtained dataset is not a separable classification problem. On the other hand, in Figure 4 (c), the KNN approach based on MI provides an imputed dataset which makes the classification stage easier, without distorting the input data distribution. This advantage is clearer for higher percentages of missing values, as it is shown in Figures 4 (b)-(d).

Finally, it has been shown that the KNN imputation procedure and its extensions provide robust tools for missing data estimation (Batista & Monard, 2003; García-Laencina *et al.*, 2008; Troyanskaya *et al.*, 2001). Its major drawback is that whenever the KNN method looks for the most similar instances, the algorithm has to search through all the data set. This limitation can be critical for large databases.

Self-Organizing Maps Imputation

A SOM is a neural network model made out of a set of nodes (or neurons), which are organized on a 2D grid and fully connected to the input layer (Kohonen, 2006). Each node has a specific topological

Figure 4. Missing data imputation in a separable two dimensional dataset (two classes: squares and circles) by two KNN methods (standard and MI-feature weighted) with $K=5$. In (a) and (b), imputed datasets by the standard approach for 10% and 20% of missing data in the second attribute are show. Meanwhile, (c) and (b) shows the imputed datasets by the MI-based procedure for the same percentages. An horizontal line denotes an incomplete pattern with a missing value in x_2 .



position in the grid, and a vector of weights of the same dimension as the input vectors. The training of a basic SOM is made by an iterative process which consists in slightly moving the nodes in the data definition space, according to the data distribution. After the weight vectors are initialized, they are updated using all input training vectors. For each iteration step and for each training pattern, the Best Matching Unit (BMU) is obtained as the closest node to the input vector according to a distance metric (usually the Euclidean distance). The weight adjustment is performed while taking into account the neighboring relations between nodes in the map. For a complete description of the SOM properties and applications, see (Kohonen, 2006).

When an incomplete input vector is given as input to a SOM (Samad & Harp, 1992), the missing variables are simply ignored when distances between observation and nodes are computed. This principle is applied both for selecting the image-node and for updating weights. Because the same variable(s) is ignored in each distance calculation (over which the minimum is taken for obtaining the BMU), it can be a valid solution. After the SOM model has been trained, it can be used to estimate missing values. When an incomplete observation is presented to the SOM, the missing input variables are ignored during the selection of the BMU. The incomplete data are imputed by the feature values of the BMU in the missing dimensions. The imputation process can be described as follows:

1. Presentation of an incomplete observation on the input layer;
2. Selection of the BMU by minimizing the distance between observation and nodes. Missing components are handled by simply excluding them from the distance calculation.
3. The replacement value for a missing item in the incomplete pattern is taken as the value for that item in the corresponding BMU.

In (Samad & Harp, 1992), this approach is compared with 1NN and standard Multi-Layer Perceptron (MLP) based imputation, and it is concluded that SOM works better than the other two methods, emphasizing that SOM-based method requires less learning observations than other models, like MLP, incomplete observations can also be used during the training stage. Following this idea, Piela (2002) implements missing data imputation in a Tree Structured Self-Organizing Map (TS-SOM), which is made of several SOMs arranged to a tree structure. The major advantages of this approach over the basic SOM are its faster convergence and its computational benefit when the number of input vector is large.

Performance Measures on Imputation Methods

The performance and capabilities of the different imputation approaches in classification problems can be evaluated considering the two kinds of tasks to be solved, i.e. *classification* and *imputation* tasks. In the first case, once the missing values have been imputed, a classifier is trained and its accuracy is measured computing the *Classification Error Rate* (CER) over the test patterns (Duda *et al.*, 2000; Bishop, 1995). Whereas, for measuring the quality of the missing data estimation, it is needed to insert incomplete data in an artificial way for different missing data percentages and different combination of attributes. Let us consider that \tilde{x}_i denotes the imputed version of the i -th attribute, and \hat{x}_i denotes the true version of the same variable. Two different criteria can be used for comparing the imputation methods,

- *Predictive Accuracy* (PAC). An imputation method should preserve the true values as far as possible. Considering that the i -th attribute has missing values in some input patterns, its imputed

version \tilde{x}_i must be close to the \hat{x}_i (variable with true values). The Pearson correlation between \tilde{x}_i and \hat{x}_i provides a good measure of the imputation performance, and its it given by:

$$PAC \equiv r = \frac{\sum_{n=1}^N (\tilde{x}_{i,n} - \bar{\tilde{x}}_i)(\hat{x}_{i,n} - \bar{\hat{x}}_i)}{\sqrt{\sum_{n=1}^N (\tilde{x}_{i,n} - \bar{\tilde{x}}_i)^2 (\hat{x}_{i,n} - \bar{\hat{x}}_i)^2}} \quad (13)$$

where $\tilde{x}_{i,n}$ and $\hat{x}_{i,n}$ denotes, respectively, the n -th value of \tilde{x}_i and \hat{x}_i , and besides, $\bar{\tilde{x}}_i$ and $\bar{\hat{x}}_i$ denotes, respectively, the mean of the N values included in \tilde{x}_i and \hat{x}_i . A good imputation method will have a value for the Pearson correlation close to 1.

- *Distributional Accuracy* (DAC). An imputation method should preserve the distribution of the true values. One measure of the preservation of the distribution of the true values is the distance between the empirical distribution function for both the imputed and the true values. The empirical distribution functions, $F_{\hat{x}_i}$ for the cases with true values, and $F_{\tilde{x}_i}$ for the cases with imputed values, are defined as

$$F_{\hat{x}_i}(x) = \frac{1}{N} \sum_{n=1}^N I(\hat{x}_{i,n} \leq x) \quad (14)$$

$$F_{\tilde{x}_i}(x) = \frac{1}{N} \sum_{n=1}^N I(\tilde{x}_{i,n} \leq x) \quad (15)$$

where I is the indicator function. The distance between these functions can be measured using the Kolmogorov-Smirnov distance, D_{KS} , which is given by

$$DAC \equiv D_{KS} = \max_n \left(\left\| F_{\hat{x}_i}(x_n) - F_{\tilde{x}_i}(x_n) \right\| \right) \quad (16)$$

where the x_n values are the jointly ordered true and imputed values of attribute x_i . A good imputation method will have a small distance value.

In many scenarios, the *CER* is the most significant metric due to the fact that the main objective is to solve a classification problem, and the imputation is a secondary task whose aim is to provide imputed values which help to solve the classification problem. For example, given two or more imputation methods and a neural network classifier, the best imputation approach will be that method which provides better *CER* in the test set given the same weight initialization for the classifier.

MODEL BASED PROCEDURES AND EXPECTATION-MAXIMIZATION ALGORITHM

The methods described in this section are called model-based methods since the researcher must make assumptions about the joint distribution of all the variables in the model. One of the most used approaches in this category is the Mixture Models trained with the Expectation-Maximization (EM) algorithm. The EM algorithm is an efficient iterative procedure to compute the Maximum Likelihood (ML) estimate in the presence of missing or hidden data (Ghahramani & Jordan, 1994a; Ghahramani & Jordan, 1994b; McLachlan & Krishnan, 1997). In ML estimation, we wish to estimate the model parameter(s) for which the observed data is the most likely. In addition to this, mixture models provide a general semi-parametric model for arbitrary densities, where the density functions are modeled as a linear combination of component densities (in a non-parametric kernel-based approach, it is a linear combination of kernel or basis functions). In particular, real valued data can be modeled as a mixture of Gaussians; and for discrete valued data, it can be modeled as a mixture of Bernoulli densities or as a mixture of multinomial densities. The EM algorithm has been successfully applied to a variety of problems involving incomplete data (Ghahramani & Jordan, 1994a; Ghahramani & Jordan, 1994b), such as training Gaussian Mixture Models (GMMs). EM associates a given incomplete-data problem with a simpler complete-data problem, and iteratively finds the maximum likelihood estimates of the missing data. The EM approach asks the question, “What parameter estimates are most likely given the data that were observed?”. For answering this question, the parameters of interest are estimated from the available data. Then these parameters are used to estimate possible values for the data that are missing. Next, the parameters are re-estimated using the available data and estimated values. Once the parameters have been obtained, the missing values are re-estimated based on the new parameters. This iterative process continues until the convergence of the estimated. In a typical situation, EM converges monotonically to a fixed point in the state space, usually a local maximum.

Each iteration of the EM algorithm consists of two processes: the E-step, and the M-step. The Expectation or E-step computes the log likelihood of the data, and the Maximization or M-step finds the parameters that maximize this likelihood [46]. Applying EM to an incomplete-data problem can actually be perceived as a special instance of local search. For example, when learning a mixture of Gaussians (Ghahramani & Jordan, 1994a; Ghahramani & Jordan, 1994b), the state space consists of all the possible assignments to the means, covariance matrices, and prior probabilities of the Gaussian distributions. Missing features can be treated naturally in this framework, by computing the expectation of the sufficient statistics for the E-step over the missing values as well as the mixture components. Thus, in the expectation, or E-step, the missing data is estimated given the observed data and current estimate of the model parameters. This is achieved using the conditional expectation. In the M-step, the likelihood function is maximized under the assumption that the missing data is known. The estimates of the missing data from the E-step are used instead of the actual missing data. The EM algorithm starts with some assignment to these state variables; for example, the mean vectors are usually initialized with k -means, and iteratively update the state variables until it converges to a state with a locally maximum likelihood estimate of training data. Convergence is assured since the algorithm is guaranteed to increase the likelihood at each iteration (McLachlan & Krishnan, 1997). The key idea that differentiates EM algorithms from any other iterative algorithms is that, missing values themselves are not necessarily estimated by the EM. Instead, the EM only finds the conditional expectations of the missing data using the observed

and the estimated parameters. However, EM may converge rather slowly and, moreover, it may not converge to the global maximum of the likelihood.

In a classification problem, the mixture-modelling framework can model the class label as a multinomial variable, i.e., the mixture model estimates the joint probability that an input vector has determined attributes and belongs to a determined class. Once the model is obtained, the most likely label for a particular input pattern may be obtained by computing the class-posterior probabilities using the Bayes theorem (Duda *et al.*, 2000; Ghahramani & Jordan, 1994a; Ghahramani & Jordan, 1994b).

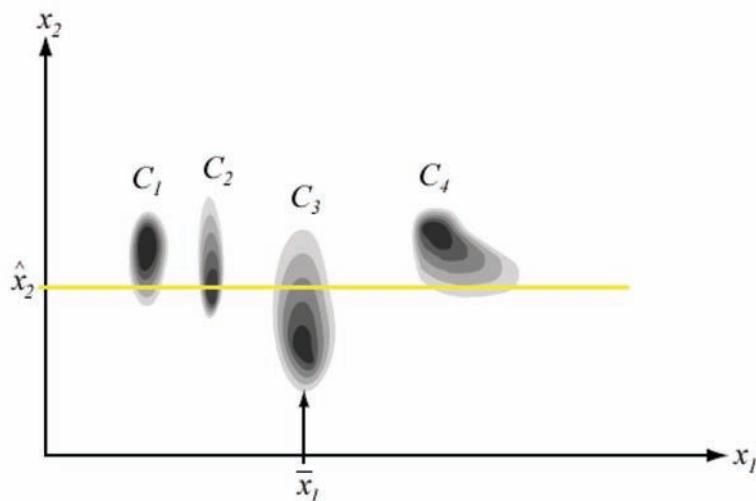
Marginalization

In addition to the mixture modeling technique with the EM algorithm, other machine learning approaches for classifying incomplete input vectors can be developed by incorporating the input data distribution into the classifier using the concept of *marginalization* (Ahmad & Tresp, 1993; Duda *et al.*, 2000). Suppose we develop a classifier by using uncorrupted data, but a test pattern is incomplete. How can we classify this incomplete pattern to obtain a minimum error? For example, consider a classifier with two input features, such that one of the features is missing for a particular pattern \mathbf{x} to be classified. Figure 5 illustrates a four-class problem, where for the test object \mathbf{x} the feature x_1 is missing.

The measured value of x_2 for \mathbf{x} is \hat{x}_2 . Clearly, if we assume that the missing value can be substituted by the mean of x_1 , \mathbf{x} will be classified as C_3 . However, if the priors are equal, class C_2 would be a better decision, because $p(\hat{x}_2 | C_2)$, estimated on the training set is the largest of the four likelihoods.

To clarify this concept, consider that a neural network, or another machine learning tool, has been successfully trained using a complete classification dataset, and it provides the class posterior probabilities given \mathbf{x} , i.e., $P(t=c|\mathbf{x})$. Suppose that for a particular test pattern \mathbf{x} , one or more input features are

Figure 5. Class conditional distributions for a four-class problem. If a test pattern misses the first feature value the optimal classification decision will be C_2 because $p(\hat{x}_2 | C_2)$ (estimated on the training set) is the largest.



unknown, $\mathbf{x} = [\mathbf{x}^o, \mathbf{x}^m]$. In this situation, the class posterior given the observed features can be computed using the Bayes rule by,

$$\begin{aligned} P(t = c | \mathbf{x}^o) &= \frac{p(t = c, \mathbf{x}^o)}{p(\mathbf{x}^o)} = \frac{\int p(t = c, \mathbf{x}^o, \mathbf{x}^m) d\mathbf{x}^m}{p(\mathbf{x}^o)} = \frac{\int P(t = c | \mathbf{x}^o, \mathbf{x}^m) p(\mathbf{x}^o, \mathbf{x}^m) d\mathbf{x}^m}{p(\mathbf{x}^o)} = \\ &= \frac{\int P(t = c | \mathbf{x}) p(\mathbf{x}) d\mathbf{x}^m}{\int p(\mathbf{x}) d\mathbf{x}^m} = \frac{\int f_c(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}^m}{\int p(\mathbf{x}) d\mathbf{x}^m}, \end{aligned} \quad (17)$$

where $f_c(\mathbf{x}) = f_c(\mathbf{x}^o, \mathbf{x}^m) = P(t = c | \mathbf{x})$ is the discriminant function of the trained classifier for the class c . In this equation, the full joint distribution $p(t = c, \mathbf{x}^o, \mathbf{x}^m)$ is marginalized over the variables \mathbf{x}^m , i.e., the decision making process is done by integrating (marginalize) the posterior probability over the unknown attributes. If the integral cannot be solved analytically, it can be done by a numerical approximation using the Monte Carlo method. Finally, we use the Bayes decision rule on the obtained posterior probabilities for classifying the incomplete test pattern.

In addition to this, efficient neural network approaches for handling missing data have been developed to incorporate the input data distribution modeling into the neural network (Ahmad & Tresp, 1993; Tresp *et al.*, 1993; Tresp *et al.*, 1994; Williams *et al.*, 2007). Ahmad and Tresp (1993) discuss Bayesian techniques for extracting class probabilities given partial data. Considering that the classifier outputs are good estimates of the class-posterior probabilities given the input vector, and it can be split up into a vector of complete features and other with unknown features, they demonstrate that the optimal solution involves integrating over the missing dimensions weighted by the local probability densities. Moreover, in (Ahmad & Tresp, 1993), closed-form approximations to the class probabilities are obtained by using Gaussian basis function networks, and they extend to an MLP classifier calculating the integrals with Monte Carlo techniques. Instead of doing a numerical approximation of these integrals, Tresp *et al.* (1993) propose an efficient solution using GMMs and Parzen windows to estimate the conditional probability densities that appear in the integral. For more details, see (Tresp *et al.*, 1993; Tresp *et al.*, 1994). As it can be observed, in addition to the class posterior probabilities estimated by a machine learning system, this procedure also requires to the estimation of the input data density, which is not explicitly available when a neural network is used as classifier. Due to this, the mixture modeling approach trained using the EM algorithm is a better solution for incomplete data classification, because it estimates the input data density and the class posterior at the same time. Recently, Williams *et al.* (2007) introduce a scheme for incomplete data classification in a probit regression model that closely approximates logistic regression. The advantage of the probit link function is that it allows an approximation to analytically integrating over missing data with respect to an auxiliary Gaussian mixture feature space model (Williams *et al.*, 2007). However, its major drawback is the restriction to linear classifier.

EMBEDDED MACHINE LEARNING SOLUTIONS: NON IMPUTATION IS REQUIRED

Up to now, this chapter has discussed several machine learning procedures where missing input attributes are replaced by plausible values and afterwards that classification is done with another machine, or the

input data density is estimated using a mixture model trained by the EM algorithm, and afterwards a test pattern is classified using the Bayes decision rule. Conversely, other approaches have been proposed for handling missing data in classification problems. Now, we will summarize the most representative machine learning procedures which are able to deal with unknown values avoiding explicit imputations, such as ensemble methods or reduced models, decision trees, and fuzzy approaches.

Classification in Sub-Space: Reduced Models

Perhaps the most straightforward procedure for dealing with missing values is to learn a different classifier for each pattern of observed values, i.e., for each combination of incomplete features. Sharpe and Solly (1995) study the diagnosis of thyroid disease with MLPs under this framework, which they refer as the network reduction approach. The advantage of this procedure is that standard classification methods can be applied to learn each possible combination of complete features. Sharpe and Solly (1995) show that learning one MLP classifier for each subspace of observed attributes led to better classification performance than using MLP imputation combined with a single MLP classifier taking all features as inputs. As an example, consider the four dimensional dataset which has been previously described in the MLP imputation method. In this scenario, a different model learns the classification task in each possible combination of complete features, i.e., in each sub-space. The classifiers are denoted by $q_{x_i}(\cdot)$, where the sub-index indicates that the attribute x_i will be discarded for performing the classification task. And thus, three different classifiers are required:

$$\mathbf{m} = [0, 1, 0, 0] \Rightarrow q_{x_2}(x_1, x_3, x_4) \quad (18)$$

$$\mathbf{m} = [0, 0, 1, 0] \Rightarrow q_{x_3}(x_1, x_2, x_4) \quad (19)$$

$$\mathbf{m} = [0, 1, 1, 0] \Rightarrow q_{x_2, x_3}(x_1, x_4) \quad (20)$$

Each classifier has to be modeled using its specific training set. The main drawback of the reduced model approach is that the number of different patterns of missing features is exponential in the number of features, and it requires a huge number of models when multiple combinations of incomplete attributes are presented. In (Sharpe & Solly, 1995), the dataset only contains four inputs, and only four different combinations of missing features, making the entire approach feasible.

Other possible option is to form an ensemble of one-class classifiers trained on each feature (Juszczak & Duin, 2004). Thus when any feature values are missing for a data point to be labeled, the ensemble can still make a reasonable decision based on the remaining classifiers.

Using Missing Data Indicators

An alternative to subspace classification is to augment the input to a standard classifier with a vector of missing data indicators. The input representation $\hat{\mathbf{x}}_n = [\mathbf{x}_n \cdot \mathbf{m}_n, \mathbf{m}_n]$ can be thought of as encoding for \mathbf{x}_n^o . Here \cdot denotes elementwise multiplication. A trained classifier can be thought of as computing

a decision function based on the observed part. In MLPs and some kernel-based classifiers, substituting \bar{x}_n for x_n is the only modification required.

Decision Trees

In this kind of methods, we stand out three well-known approaches: ID3, C4.5 and CN2. These procedures can handle missing values in any attribute for both training and test sets (Clark & Niblett, 1982; Quinlan, 1986; Quinlan, 1989; Quinlan, 1993; Zheng & Low, 1999). ID3 is a basic top-down decision tree algorithm that handles an unknown attribute by generating an additional edge for the unknown. Thus, an unknown edge has been taken as a new possible value for each attribute and it has been treated in the same way as other values. C4.5 is an extension of ID3 proposed by Quinlan (1993). It uses a probabilistic approach to handle missing values in the training and test data set. In this approach, the way of dealing with missing values changes during the training and testing stages. During the training phase, each value for an attribute is assigned a weight (Quinlan, 1989; Quinlan 1993; Zheng & Low, 1999). If an attribute value is known, then the weight is equal to one; otherwise, the weight of any other value for that attribute is the relative frequency of that attribute. On the testing phase, if a test case is incomplete, it explores all the available branches (below the current node) and decides the class label by the most probabilistic value. The CN2 is an algorithm for inducing propositional classification rules (Clark & Niblett, 1982). It uses a rather simple imputation method to treat missing data. Every missing value is filled in with its attribute most common known value, before computing the entropy measure.

Fuzzy Approaches

Many fuzzy procedures have been developed in order to handle missing data. Ishibuchi *et al.* (1993) propose an MLP classification system where unknown values are represented by interval inputs. For example, if the pattern space of a particular classification problem is the d -dimensional unit cube $[0, 1]^d$, each unknown feature value is represented by the interval input that includes all the possible values of that attribute. When the attribute value is completely known, it is also represented by intervals (e.g., 0.3 is represented by $[0.3, 0.3]$). This network is trained by means of back-propagation algorithm for fuzzy input vectors [76]. Gabrys (2000) develops a General Fuzzy Min-Max (GFMM) neural network using hyperbox fuzzy sets. A hyperbox defines a region of the d -dimensional pattern space, by its min-point and its max-point, so the patterns contained within the hyperbox have “full class” membership. Learning in the GFMM neural network for classification consists of creating and adjusting hyperboxes in pattern space (Gabrys, 2000; Gabrys, 2002). This procedure handles missing values in a similar way as the interval inputs, i.e., the missing attribute is modeled by a real valued interval spanning the whole range of values. In (Berthold & Huber, 1998; Nauck & Kruse, 1997), some techniques to tolerate missing values based on a system of fuzzy rules for classification are proposed. In general, a fuzzy rule based classifier consists of a set of rules for each possible category, where each rule can be decomposed into individual one-dimensional membership functions corresponding to the fuzzy sets (Berthold & Huber, 1998). When an incomplete pattern has to be classified, the rules are obtained using only one-dimensional membership function of the known attributes, which is computationally very efficient.

FUTURE TRENDS AND CONCLUSION

Missing or incomplete data is a usual drawback in many real-world applications of pattern classification. Data may contain unknown features due to different reasons, e.g., sensor failures producing a distorted or unmeasurable value, data occlusion by noise, non-response in surveys. This chapter reviews several machine learning methods for incomplete pattern classification. In general, pattern classification with missing data concerns two different problems, *handling missing values* and *pattern classification*. Depending on how both problems are solved, most of the approaches in the literature can be grouped into four different types of approaches. First, the easiest way for dealing with missing data is the deletion of incomplete values, being the loss of information its main disadvantage. The second kind of methods is imputation, i.e. to estimate and fill in the unknown input data. In this category, we can distinguish between statistical procedures, such as mean imputation or multiple imputation, and machine learning approaches, such as imputation with neural networks. Other well-known approaches are based on the estimation of the input data distribution, being the EM algorithm the most remarkable procedure. Most of these solutions work well in many situations, being a recommended way to handle missing data. The last methods analyzed in this work are embedded machine learning solutions that are able to handle missing values during the training process without an explicit imputation, such as reduced approaches or decision trees.

It is evident that the research direction can still be broadened as there are some open questions need to be answered, such as “can an efficient imputation solution enhance the decision making process?”. In classification tasks with missing values, the main objective of an imputation method must be to help to enhance the classification accuracy. In the recent years, some works have been developed in order to provide missing data estimation focused on solving the classification problem, and not on obtaining imputed values which minimize the error between the predictions and the “true” values. Since it remains quite difficult to compare different techniques, some pending questions to be answered are to establish an unified evaluation criteria and an open access repository for incomplete datasets. First, the most important aspect to be evaluated in incomplete data classification problems is the missing data influence on the classification error rate. Moreover, imputation accuracy can also be evaluated in solutions based on missing data estimation. In this kind of approaches, different missing data percentages can artificially be inserted and two different criteria can be measured: predictive and distributional accuracy. An imputation method should preserve the true values and the input data distribution (as far as possible). Up to now, an unified criterion to compare different missing data solutions has to be established because most of the current research works use different evaluation criteria. Aspect of running time and memory requirements have also to be considered for comparison. The requirements for speed vary from one application to the other. Missing values in fast moving data such as the stock market need to be estimated in the shortest possible time, whereas this requirement is less important for an opinion poll. Another important topic is to develop a database for incomplete pattern classification. There is not an open access database to evaluate the usefulness of the different methods. This database should have to be composed of real and artificial datasets in several real-life classification scenarios. In the last type of datasets, they must include different missing data percentages (where the true values are known) in several attributes (relevant and irrelevant features to classification).

The research works on incomplete data classification is currently grows wider and it is well known how useful and efficient are most of solutions based on machine learning. The correct choice of a missing data treatment is a hard and complex task, i.e., a method can work well in some problems, and in contrast,

its results are not good in other applications. Thus, a previous analysis of the classification problem to be solved is very important in order to select the most suitable missing data treatment.

REFERENCES

- Ahmad, S., & Tresp, V. (1993). Some solutions to the missing feature problem in vision. [San Mateo, CA: Morgan Kaufmann Publishers Inc.]. *Advances in Neural Information Processing Systems*, 5, 393–400.
- Allison, P. D. (2001). *Missing data*. Newbury Park, CA: Sage.
- Batista, G., & Monard, M. C. (2002). A study of k-nearest neighbour as an imputation method. In *Proceedings of the Soft Computing Systems - Design, Management and Applications, HIS 2002*, Santiago, Chile (pp. 251-260).
- Batista, G., & Monard, M. C. (2003). *Experimental comparison of K-nearest neighbour and mean or mode imputation methods with the internal strategies used by C4.5 and CN2 to treat missing data* (Tech. Rep.). University of Sao Paulo.
- Berthold, M. R., & Huber, K. P. (1998). Missing values and learning of fuzzy rules. *International Journal of Uncertainty . Fuzziness and Knowledge-Based Systems*, 6(2), 171–178. doi:10.1142/S021848859800015X
- Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford, UK: Oxford University Press.
- Caruana, R. (1997). *Multitask learning*. Unpublished doctoral dissertation, Carnegie Mellon University.
- Clark, P., & Niblett, T. (1982). The CN2 induction algorithm. *Machine Learning*, 3(4), 261–283.
- Cooke, M., Green, P., & Crawford, M. (1994). Handling missing data in speech recognition. In *Proceedings of the International Conference on Spoken Language Processing*, Yokohama, Japan (pp. 1555-1558).
- DiCesare, G. (2006). *Imputation, estimation and missing data in finance*. Unpublished doctoral dissertation, University of Waterloo.
- Duda, R. O., Hart, P. E., & Stork, D. G. (2000). *Pattern classification*. New York: Wiley-Interscience.
- Gabrys, B. (2000). Pattern classification for incomplete data. In *Proceedings of the International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, Brightom, UK (pp. 454-457).
- Gabrys, B. (2002). Neuro-fuzzy approach to processing inputs with missing values in pattern recognition problems. *International Journal of Approximate Reasoning*, 30(3), 149–179. doi:10.1016/S0888-613X(02)00070-1
- García-Laencina, P. J., Figueiras-Vidal, A. R., Serrano-García, J., & Sancho-Gómez, J. L. (2005). Exploiting multitask learning schemes using private subnetworks. In J. Cabestany, et al. (Ed.), *Computational intelligence bioinspired systems* (LNCS 3512, pp. 233-240). Berlin, Germany: Springer.

- García-Laencina, P. J., Sancho-Gómez, J. L., Figueiras-Vidal, A. R., & Verleysen, M. (2008). Exploiting multitask learning schemes using private subnetworks. In *Proceedings of the European Symposium on Artificial Neural Networks*, Bruges, Belgium (pp. 37-42).
- García-Laencina, P. J., Serrano, J., Figueiras-Vidal, A. R., & Sancho-Gómez, J. L. (2007). Multi-task neural networks for dealing with missing inputs. In J. Mira & J. R. Álvarez (Eds.), *Proceedings of the International Work Conference on the Interplay between Natural and Artificial Computation*, Murcia, Spain (LNCS 4527, pp. 282–291). Berlin, Germany: Springer.
- Ghahramani, Z., & Jordan, M. I. (1994a). *Learning from incomplete data* (Tech. Rep. AIM-1509). Massachusetts Institute of Technology, Cambridge, MA, USA.
- Ghahramani, Z., & Jordan, M. I. (1994b). Supervised learning from incomplete data via an EM approach. In J. D. Cowan, et al. (Ed.), *Advances on neural information processing systems 6*. (pp. 120-127). San Francisco: Morgan Kaufmann Publishers Inc.
- Gupta, A., & Lam, M. S. (1996). Estimating missing values using neural networks. *The Journal of the Operational Research Society*, 47, 229–238.
- Halatchev, M., & Gruenwald, L. (2005). Estimating missing values in related sensor data streams. In *Proceedings of the International Conference on Management of Data*, Goa, India (pp. 83-94).
- Ishibuchi, H., Miyazaki, A., Kwon, K., & Tanaka, H. (1993). Learning from incomplete training data with missing values and medical application. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, Nagoya, Japan (pp. 1871-1874).
- Jain, A. K., Duin, R. P. W., & Mao, J. (2000). Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), 4–37. doi:10.1109/34.824819
- Jerez, J. M., Molina, I., Subirats, J. L., & Franco, L. (2006). Missing data imputation in breast cancer prognosis. In *Proceedings of the IASTED International Multiconference in Biomedical Engineering*, Innsbruck, Austria (pp. 323-328).
- Ji, C., & Elwalid, A. (2000). Measurement-based network monitoring: missing data formulation and scalability analysis. In *Proceedings of the IEEE International Symposium on Information Theory*, Sorrento, Italy (pp. 78).
- Jian, K., Chen, H., & Yuan, S. (2005). Classification for incomplete data using classifier ensembles. In *Proceedings of the International Conference on Neural Networks and Brain*, Beijing, China (pp. 559-563).
- Juszczak, P., & Duin, R. P. W. (2004). Combining one-class classifiers to classify missing data. In F. Roli, et al. (Ed.), *Multiple classifier systems* (LNCS 3077, pp. 92-101). Berlin, Germany: Springer.
- Kim, H., Golub, G. H., & Park, H. (2004). Imputation of missing values in DNA microarray gene expression data. In *Proceedings of the IEEE Computational Systems Bioinformatics Conference*, Standford, CA, USA (pp. 572-573).
- Kohonen, T. (2006). *Self-organizing maps* (3rd ed.). Berlin, Germany: Springer.

- Krause, S., & Polikar, R. (2003). An ensemble of classifiers for the missing feature problem. In *Proceedings of the International Joint Conference on Neural Networks*, Portland, USA (pp. 553-558).
- Kwak, N., & Choi, C.-H. (2002). Input feature selection by mutual information based on parzen window. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12), 1667–1671. doi:10.1109/TPAMI.2002.1114861
- Lakshminarayan, K., Harp, S. A., & Samad, T. (2004). Imputation of missing data in industrial databases. *Applied Intelligence*, (11): 259–275.
- Little, R. J. A., & Rubin, D. B. (2002). *Statistical analysis with missing data* (2nd ed.). New York: John Wiley & Sons.
- Liu, P., El-Darzi, E., Lei, L., Vasilakis, C., Chountas, P., & Huang, W. (2005). An analysis of missing data treatment methods and their application to health care dataset. In X. Li, et al. (Eds.), *Advance data mining applications* (LNCS 3584, pp. 583-590). Berlin, Germany: Springer.
- Markey, M. K., & Patel, A. (2004). Impact of missing data in training artificial neural networks for computer-aided diagnosis. In *Proceedings of the International Conference on Machine Learning Applications*, Louisville, KY (pp. 351-354).
- McLachlan, G. J., & Krishnan, T. (1997). *The EM algorithm and extensions*. New York: Wiley.
- Mohammed, H. S., Stepenosky, N., & Polikar, R. (2006). An ensemble technique to handle missing data from sensors. In *Proceedings of the IEEE Sensor Applications Symposium*, Houston, TX, USA (pp. 101-105).
- Nauck, D., & Kruse, R. (1999). Learning in neuro-fuzzy systems with symbolic attributes and missing values. In *Proceedings of the International Conference on Neural Information Processing*, Perth, WA, Australia (pp. 142-147).
- Nguyen, L. N., & Scherer, W. T. (2003). *Imputation techniques to account for missing data in support of intelligent transportation systems applications* (Tech. Rep.). University of Virginia, USA.
- Nordbotten, S. (1996). Neural network imputation applied to the Norwegian 1990 population census data. *Journal of Official Statistics*, 12, 385–401.
- Parveen, S. (2003). *Connectionist approaches to the deployment of prior knowledge for improving robustness in automatic speech recognition*. Unpublished doctoral dissertation, University of Sheffield.
- Piela, P. (2002). Introduction to self-organizing maps modelling for imputation - techniques and technology. *Research in Official Statistics*, 2, 5–19.
- Proschan, M. A., McMahon, R. P., Shih, J. H., Hunsberger, S. A., Geller, N., Knatterud, G., & Wittes, J. (2001). Sensitivity analysis using an imputation method for missing binary data in clinical trials. *Journal of Statistical Planning and Inference*, 96(1), 155–165. doi:10.1016/S0378-3758(00)00332-3
- Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1), 81–106.
- Quinlan, J. R. (1989). Unknown attribute values in induction. In Proceedings of the *International Workshop on Machine Learning* (pp. 164-168). San Francisco: Morgan Kaufmann Publishers Inc.

- Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco: Morgan Kaufmann.
- Ripley, B. D. (1996). *Pattern recognition and neural networks*. New York: Cambridge University Press.
- Rubin, D. B. (1987). *Multiple imputation for nonresponse in surveys*. New York: Wiley.
- Samad, T., & Harp, S. A. (1992). Self-organization with partial data. *Network*, 3, 205–212.
- Sancho-Gómez, J. L., García-Laencina, P. J., & Figueiras-Vidal, A. R. (2008). Combining missing data imputation and classification in a multi-layer perceptron. *Intelligent Automation and Soft Computing*.
- Schaffer, J. L. (1997). *Analysis of incomplete multivariate data*. Boca Raton, FL: Chapman & Hall.
- Sharpe, I. G., & Kofman, P. (2003). Using multiple imputation in the analysis of incomplete observations in finance. *Journal of Financial Econometrics*, 1(2), 216–249. doi:10.1093/jjfinec/nbg013
- Sharpe, P. K., & Solly, R. J. (1995). Dealing with missing values in neural network-based diagnostic systems. *Neural Computing & Applications*, 3(2), 73–77. doi:10.1007/BF01421959
- Silver, D. L. (2000). *Selective transfer of neural network task knowledge*. Unpublished doctoral dissertation, University of Western Ontario.
- Tresp, V., Ahmad, S., & Neuneier, R. (1993). Training neural networks with deficient data. In J. D. Cowan, et al. (Eds.), *Advances on neural information processing systems 6* (pp. 128-135). San Francisco: Morgan Kaufmann Publishers Inc.
- Tresp, V., Neuneier, R., & Ahmad, S. (1994). Efficient methods for dealing with missing data in supervised learning. In G. Tesauro, et al. (Eds.), *Advances on neural information processing systems 7* (pp. 689-696). Cambridge, MA: The MIT Press.
- Troyanskaya, O., Cantor, M., Alter, O., Sherlock, G., Brown, P., & Botstein, D. (2001). Missing value estimation methods for DNA microarrays. *Bioinformatics (Oxford, England)*, 17(6), 520–525. doi:10.1093/bioinformatics/17.6.520
- Wang, L., & Fan, X. (2004). Missing data in disguise and implications for survey data analysis. *Field Methods*, 16(3), 332–351. doi:10.1177/1525822X03262276
- Williams, D., Liao, X., Xue, Y., Carin, L., & Krishnapuram, B. (2007). On classification with incomplete data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3), 427–436. doi:10.1109/TPAMI.2007.52
- Yoon, S. Y., & Lee, S. Y. (1999). Training algorithm with incomplete data for feed-forward neural networks. *Neural Processing Letters*, 10, 171–179. doi:10.1023/A:1018772122605
- Zheng, Z., & Low, B. T. (1999). Classifying unseen cases with many missing values. In N. Zhong & L. Zhou (Eds.), *Proceedings of the Third Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining*, Beijing, China (LNAI 1574, pp. 370-374). Berlin, Germany: Springer.

KEY TERMS AND DEFINITIONS

Pattern: is an entity that can be represented by a set of properties and variables, which are known as features or attributes. As examples, a pattern can be a fingerprint image, a human face or a speech signal

Pattern Classification: is a scientific discipline whose aim is the classification of the objects into a set of categories or classes.

Missing Data: data are said to be missing when there is no information for one or more pattern on one or more features in a research study.

Missing Data Pattern: it describes which values are observed in the input data matrix and which values are missing.

Missing Data Indicator Matrix: it defines the missing data pattern.

Missing Data Mechanism: is the relationship between missingness and the known attributes in the input data matrix, i.e., the probability that a set of values are missing given the values taken by the observed and missing features

Missing Completely at Random (MCAR): data are MCAR when the event that a particular item is missing is independent of observable and unknown features.

Missing At Random (MAR): data are MAR when the missing data pattern is independent of all unobserved features, although it may be traceable or predictable from other variables in the database

Not Missing At Random (NMAR): data are NMAR when the missing data pattern is non-random and depends on the missing variables. In this situation, the missing variables in the NMAR case cannot be predicted only from the available variables in the database, and the missingness mechanism is informative

Listwise/Casewise analysis: it deletes the cases that have missing data for any one variable used in a particular study.

Available-Case Analysis: it only uses the cases with available features for a particular study.

Imputation: is a generic term for filling in unknown features with plausible values provided by a missing data estimator. Missing values are estimated from the available data.

Multiple Imputation: a procedure which replaces each missing value with a set of plausible values that represent the uncertainty about the right value to impute. The multiply imputed data sets are then analyzed by using standard procedures for complete data and combining the results from these analyses.

Error Rate in Classification: the proportion of patterns that have been incorrectly classified by a decision model.

Predictive Accuracy: an imputation procedure should maximise the preservation of true values. That is, it should result in imputed values that are as ‘close’ as possible to the true values

Distributional Accuracy: an imputation procedure should preserve the distribution of the true data values. That is, marginal and higher order distributions of the imputed data values should be essentially the same as the corresponding distributions of the true values

Expectation-Maximization Algorithm: is an efficient iterative procedure to compute the maximum likelihood estimates of parameters in probabilistic models, in the presence of missing or hidden data

Marginalization: is used to compensate for the missing values where the unreliable features of a pattern are integrated out of the distribution of a class, constrained by upper and lower bounds on the true values of these components implicit in their observed values. The resulting distributions with the smaller number of components are then used to compute the likelihood for that pattern

ENDNOTE

- ¹ Henceforth, the terms pattern, input vector, case, observation, sample, and example are used as synonyms.

ACRONYMS AND ABBREVIATIONS (IN ALPHABETICAL ORDER)

BMU: Best Matching Unit

BP: Back-Propagation

CAD: Computer Aided Diagnosis

CER: Classification Error Rate

DAC: Distributional Accuracy

EM: Expectation-Maximization

GFMM: General Fuzzy Min-Max

GMM: Gaussian Mixture Model

HEOM: Heterogeneous Euclidean-Overlap Metric

KNN: K Nearest Neighbours

MAR: Missing At Random

MCAR: Missing Completely At Random

MI: Mutual Information

ML: Maximum Likelihood

MLP: Multi-Layer Perceptron

MSE: Mean Square Error

MTL: Multi-Task Learning

NMAR: Not Missing At Random

PAC: Predictive Accuracy

SOM: Self-Organizing Maps

SVM: Support Vector Machines

Chapter 8

Clustering and Visualization of Multivariate Time Series

Alfredo Vellido

Universidad Politécnica de Cataluña, Spain

Iván Olier

Universidad Politécnica de Cataluña, Spain

ABSTRACT

The exploratory investigation of multivariate time series (MTS) may become extremely difficult, if not impossible, for high dimensional datasets. Paradoxically, to date, little research has been conducted on the exploration of MTS through unsupervised clustering and visualization. In this chapter, the authors describe generative topographic mapping through time (GTM-TT), a model with foundations in probability theory that performs such tasks. The standard version of this model has several limitations that limit its applicability. Here, the authors reformulate it within a Bayesian approach using variational techniques. The resulting variational Bayesian GTM-TT, described in some detail, is shown to behave very robustly in the presence of noise in the MTS, helping to avert the problem of data overfitting.

INTRODUCTION

The analysis of MTS is an established research area, and methods to carry it out have stemmed both from traditional statistics and from the Machine Learning and Computational Intelligence fields. In this chapter, we are mostly interested in the latter, but considering a mixed approach that can be ascribed to Statistical Machine Learning.

MTS are often analyzed for prediction and forecasting and, therefore, the problem is considered to be supervised. In comparison, little research has been conducted on the problem of unsupervised clustering for the exploration of the dynamics of multivariate time series (Liao, 2005). It is sensible to assume that, in many problems involving MTS, the states of a process may be reproduced or *revisited* over time; therefore, clustering structure is likely to be found in the series. Furthermore, for exploratory purposes, it

DOI: 10.4018/978-1-60566-766-9.ch008

would be useful to visualize the way these series evolve from one cluster or region of clusters to another over time, as this could provide intuitive visual cues for forecasting as well as for the distinction between mostly stable states, smooth dynamic regime transitions, and abrupt changes of signal regime.

It has been argued (Keogh and Lin, 2005) that, in some cases, time series clustering is a meaningless endeavour. It is generally agreed, though, that this might only apply to certain types of time series clustering methods, such as those resorting to subsequence clustering. Nevertheless, it has recently been shown (Simon, Lee & Verleysen, 2006) that, for univariate time series and with an adequate pre-processing based on embedding techniques, clustering using Self-Organizing Maps (SOM: Kohonen, 2001) can indeed be meaningful. This must be understood in the sense that the distribution of two different time series over the SOM map (or, in the case of the statistically principled models we present in this chapter, over the latent space of states) should be significantly different. In this chapter we analyze multivariate time series but, equally, clustering would be meaningful only if the distribution of two multivariate time series over the latent space of states clearly differed (which is the case).

One of the most commonly used Machine Learning methods for the clustering of MTS is indeed SOM, in general without accounting for the violation of the independent identically distributed (i.i.d.) condition. Several extensions of SOM have been developed to explicitly accommodate time series through recurrent connectivity (Chappell & Taylor, 1993; Strickert & Hammer, 2005; Tiño, Farkas & van Mourik, 2006; Voetglin, 2002). The SOM was originally defined as a biologically inspired model but has long ago veered away towards general data analysis. Despite attempts to fit it into a probabilistic framework (e.g., Kostiainen & Lampinen, 2002; Yin & Allinson, 2001), it has mostly retained its heuristic definition, which is at the origin of some of its limitations. Generative Topographic Mapping (GTM: Bishop, Svensén & Williams, 1998a) is a nonlinear latent variable model that was originally devised as a probabilistic alternative to SOM that aimed to overcome aforementioned limitations. Its probability theory foundations have enabled the definition of principled extensions for hierarchical structures (Tiño & Nabney, 2002), missing data imputation (Carreira-Perpiñan, 2000; Vellido, 2006), adaptive regularization (Bishop, Svensén & Williams, 1998b; Vellido, El-Deredy & Lisboa, 2003), discrete data modelling (Bishop, Svensén & Williams, 1998b; Girolami, 2002), and robust outlier detection and handling (Bullen, Cornford & Nabney, 2003; Vellido & Lisboa, 2006), amongst others.

The GTM Through Time (henceforth referred to as GTM-TT: Bishop, Hinton & Strachan, 1997) is one such extension of GTM for the analysis of MTS. It explicitly accounts for the violation of the i.i.d. condition, given that it is defined as a constrained hidden Markov model (HMM: Rabiner, 1989). Its original formulation has several limitations, including the lack of adequate regularization procedures, which makes it prone to be negatively affected by the presence of uninformative noise in the MTS. In its basic formulation, the GTM is trained within the Maximum Likelihood (ML) framework using the Expectation-Maximization (EM) algorithm, and overfitting may occur unless regularization methods are applied. In (Bishop, Svensén & Williams, 1998b; Vellido, El-Deredy & Lisboa, 2003), regularization strategies based on Bayesian evidence approaches were introduced. Unfortunately, they require a number of modelling assumptions and approximations to be made. An alternative for the formulation of GTM that confers the model with regularization capabilities, while avoiding such approximations, is that of using variational techniques (Beal, 2003). A Variational GTM model based on the GTM with a Gaussian Process (GP) prior with added Bayesian estimation of its parameters was recently described in (Olier & Vellido, 2008). This Variational GTM was shown to limit the negative effect of data overfitting, while retaining the data visualization capabilities of the model. In this chapter, we describe the extension of the variational approach to the analysis of MTS, defining the Variational Bayesian GTM-TT.

Its capability to deal with the presence of uninformative noise in the MTS is illustrated through some simple experiments.

The rest of the chapter is structured as follows: The standard GTM-TT and its Gaussian process (GP) formulation are described in the next two sections, which is followed by a Bayesian reformulation that is then inserted within a variational framework for the estimation of the model parameters. The results of several illustrative experiments are then presented and discussed. The chapter wraps up with a brief conclusion section.

THE ORIGINAL GENERATIVE TOPOGRAPHIC MAPPING THROUGH TIME

D -variate MTS are sequences of D -dimensional data points that are measured at successive times, in often uniformly spaced intervals. They can also be seen as a sequence of random variables that do not comply with the independent and identically distributed (i.i.d.) condition, because they are not mutually independent. For this reason, models that are valid for i.i.d. data are not adequate for their analysis. For instance, the standard GTM would model and visually represent time series in the same way regardless the way their data points or samples were ordered. This is the unwanted result of ignoring the sequential nature of the time series. Here, we describe an extension of GTM, namely the GTM-TT, that is suited to the analysis of non-i.i.d. data.

GTM-TT can be seen as a GTM model in which the latent points are considered as hidden states and are linked by transition probabilities in a similar fashion to HMMs. Moreover, GTM-TT can be understood as a topology-constrained HMM where the emission probabilities are controlled by the GTM mixture distribution.

Assuming an observed D -variate time series $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n, \dots, \mathbf{x}_N\}$ of $n = 1 \dots N$ samples and a sequence of discrete hidden states $\{z_1, z_2, \dots, z_n, \dots, z_N\}$, where z_n is one of K possible states, the probability of the observations is given by:

$$p(\mathbf{X}) = \sum_{\text{all paths}} p(\{z_1, \dots, z_N\}, \mathbf{X}) \quad (1)$$

where $p(\{z_1, \dots, z_N\}, \mathbf{X})$ defines the complete-data likelihood as in HMM models (Rabiner, 1989) and takes the following form:

$$p(\{z_1, \dots, z_N\}, \mathbf{X}) = p(z_1) \prod_{n=2}^N p(z_n | z_{n-1}) \prod_{n=1}^N p(\mathbf{x}_n | z_n) \quad (2)$$

As in HMMs, $p(z_1)$ is the probability of the initial state z_1 , $p(z_n | z_{n-1})$ is the probability of transition from the state z_{n-1} to the state z_n , and $p(\mathbf{x}_n | z_n)$ is the probability of that the state z_n emits the observation \mathbf{x}_n .

Therefore, the parameters of the GTM-TT model are defined as follows:

$$\Theta = (\pi, \mathbf{A}, \mathbf{W}, \beta) \quad (3)$$

where:

$$\boldsymbol{\pi} = \{\pi_k\} : \pi_k = p(z_1 = k) \quad (4)$$

is a K -dimensional vector of the initial state probabilities,

$$\mathbf{A} = \{a_{jk}\} : a_{jk} = p(z_n = k \mid z_{n-1} = j) \quad (5)$$

is a matrix of $K \times K$ elements of transition state probabilities, and

$$\{\mathbf{W}, \beta\} : p(\mathbf{x}_n \mid z_n = k) = \left(\frac{\beta}{2\pi} \right)^{D/2} \exp \left(-\frac{\beta}{2} \|\mathbf{x}_n - \mathbf{y}_k(\mathbf{u}_k, \mathbf{W})\|^2 \right) \quad (6)$$

are the parameters of the emission probabilities. As in standard GTM, spherical Gaussian distributions with common inverse variance β and reference vectors $\mathbf{y}_k = \mathbf{W}\Phi(\mathbf{u}_k)$ control the emission probabilities. In GTM-TT, the hidden states are topology constrained by the discrete latent space of points \mathbf{u}_k (which is a two-dimensional space for the purpose of data visualization).

The reference vectors and the latent points are related by the generalized linear regression model of a mixture of spheric Gaussian basis functions. For mathematical convenience, it is useful defining a state in the form of a binary vector \mathbf{z}_n such that its component $z_{k,n}$ returns 1 if \mathbf{z}_n is in state k , and zero otherwise. Furthermore, \mathbf{z}_n complies with the restriction $\sum_{k=1}^K z_{k,n} = 1$.

Using this notation, the initial state probabilities, the transition state probabilities and the emission probabilities are defined as:

$$p(z_1 \mid \boldsymbol{\pi}) = \prod_{k=1}^K \pi_k^{z_{k,1}} \quad (7)$$

$$p(z_n \mid z_{n-1}, \mathbf{A}) = \prod_{j=1}^K \prod_{k=1}^K a_{jk}^{z_{k,n} z_{j,n-1}} \quad (8)$$

$$p(\mathbf{x}_n \mid z_n, \mathbf{W}, \beta) = \left(\frac{\beta}{2\pi} \right)^{D/2} \prod_{k=1}^K \left\{ \exp \left(-\frac{\beta}{2} \|\mathbf{x}_n - \mathbf{y}_k\|^2 \right) \right\}^{z_{k,n}} \quad (9)$$

Eqs. 7 to 9 lead to the definition of the complete data log-likelihood as:

$$\begin{aligned} \ln p(\mathbf{Z}, \mathbf{X} \mid \boldsymbol{\Theta}) &= \sum_{k=1}^K z_{k,1} \ln \pi_k + \sum_{n=2}^N \sum_{j=1}^K \sum_{k=1}^K z_{j,n-1} z_{k,n} \ln a_{jk} \\ &\quad + \frac{ND}{2} \ln \left(\frac{\beta}{2\pi} \right) - \frac{\beta}{2} \sum_{n=1}^N \sum_{k=1}^K z_{k,n} \|\mathbf{x}_n - \mathbf{y}_k\|^2 \end{aligned} \quad (10)$$

where \mathbf{Z} is a matrix of N column vectors \mathbf{z}_n .

Parameter estimation can be accomplished in GTM-TT by maximum likelihood using the EM algorithm. However, a straightforward calculation of Eq.1 is very inefficient. As in HMMs, a recursive forward-backward procedure (Baum & Egon, 1967) can be implemented to relieve the computational load of the GTM-TT likelihood (or log-likelihood) estimation. Details of these calculations can be found in (Vellido & Olier, 2008).

A GAUSSIAN PROCESS FORMULATION OF GENERATIVE TOPOGRAPHIC MAPPING THROUGH TIME

The original formulation of GTM-TT described in the previous section has the same restriction of the standard GTM, that is, the generalized linear regression function $\mathbf{y} = \mathbf{W}\Phi(\mathbf{u})$, introducing the number of basis functions as an additional free parameter. In this section, we extend the GP formulation of GTM to the original GTM-TT. The GTM-TT with GP formulation has the following set of parameters:

$$\Theta = (\boldsymbol{\pi}, \mathbf{A}, \mathbf{Y}, \beta) \quad (11)$$

where the definitions and calculations of $\boldsymbol{\pi}$, the initial state probabilities, and \mathbf{A} , the transition state probabilities, remain the same as in the original GTM-TT. The parameter \mathbf{Y} is a $K \times D$ matrix of centroids of K Gaussian generators and β their common inverse variance which define the emission probabilities as follows:

$$\{\mathbf{Y}, \beta\} : p(\mathbf{x}_n | z_n = k) = \left(\frac{\beta}{2\pi} \right)^{D/2} \exp \left\{ -\frac{\beta}{2} \|\mathbf{x}_n - \mathbf{y}_k\|^2 \right\} \quad (12)$$

With this, the complete-data likelihood becomes:

$$\begin{aligned} \ln p(\mathbf{Z}, \mathbf{X} | \Theta) = & \sum_{k=1}^K z_{k,1} \ln \pi_k + \sum_{n=2}^N \sum_{j=1}^K \sum_{k=1}^K z_{j,n-1} z_{k,n} \ln a_{jk} \\ & + \frac{ND}{2} \ln \left(\frac{\beta}{2\pi} \right) - \frac{\beta}{2} \sum_{n=1}^N \sum_{k=1}^K z_{k,n} \|\mathbf{x}_n - \mathbf{y}_k\|^2 \end{aligned} \quad (13)$$

Parameter estimation in GTM-TT with GP formulation can be accomplished by ML using the EM algorithm in a similar fashion to the standard GTM-TT, where the update expressions for parameters $\boldsymbol{\pi}$, \mathbf{A} and β remain the same. Parameter \mathbf{Y} is estimated as in the standard GTM using a GP for the mapping from the hidden states to the data space, by setting a prior distribution over \mathbf{Y} defined by:

$$P(\mathbf{Y}) = (2\pi)^{-KD/2} |\mathbf{C}|^{-D/2} \prod_{d=1}^D \exp \left(-\frac{1}{2} \mathbf{y}_{(d)}^T \mathbf{C}^{-1} \mathbf{y}_{(d)} \right) \quad (14)$$

where $\mathbf{y}_{(d)}$ is each of the row vectors (centroids) of the matrix \mathbf{Y} and \mathbf{C} is a matrix where each element is a covariance function that can be defined as:

$$C_{ij} = C(\mathbf{u}_i, \mathbf{u}_j) = \varepsilon \exp\left(-\frac{\|\mathbf{u}_i - \mathbf{u}_j\|^2}{2\alpha^2}\right), \quad i, j = 1 \dots K \quad (15)$$

and where parameter ε is usually fixed *a priori* to a value of 1. The α parameter controls the flexibility of the mapping from the latent space to the data space. An extended review of covariance functions can be found in (Abrahamsen, 1997). The vector $\mathbf{u}_j, j = 1 \dots K$ corresponds to the state j in a latent space of usually lower dimension than that of the data space. Thus, a topography over the states is defined by the GP as in the standard GTM.

The parameter \mathbf{Y} is estimated from:

$$(\mathbf{G} + \beta^{-1}\mathbf{C}^{-1})\mathbf{Y} = \mathbf{RX} \quad (16)$$

where \mathbf{R} is the matrix of responsibilities that can be estimated using the forward-backward procedure in a similar fashion to the standard GTM-TT; \mathbf{G} is a diagonal matrix formed by the elements $g_{kk} = \sum_n r_{k,n}$; and \mathbf{C} is the matrix of covariance functions.

BAYESIAN GTM THROUGH TIME

Although the ML framework is widely used for parameter optimization, it shows two important weaknesses: Its maximization process does not take into account the model complexity and it tends to overfit the model to the training data. The complexity in GTM-TT is related to the number of hidden states, their degree of connectivity and the dimension of the hidden space. Usually, for visualization purposes, the dimension of the hidden space is limited to be less or equal to three. The number of hidden states and the maximum number of possible state transitions are strictly correlated by a squared power. In order to avoid overfitting, researchers have commonly limited the complexity of their models by restricting the number of possible state transitions (Bishop, Hinton & Strachan, 1997) or by fixing the transition state probabilities *a priori* (Kabán & Girolami, 2002). The alternative technique of cross-validation is computationally expensive and it could require large amounts of data to obtain low-variance estimates of the expected test errors.

A more elegant solution to control overfitting and complexity is providing a Bayesian formulation for the model (MacKay, 1997; Stolcke & Omohundro, 1993). The Bayesian approach treats the parameters as unknown quantities and provides probability distributions for their priors. Bayes' theorem can then be used to infer the posterior distributions over the parameters. The model parameters can thus be considered as hidden variables and integrated out to describe the marginal likelihood as:

$$p(\mathbf{X}) = \int p(\Theta)p(\mathbf{X} | \Theta)d\Theta, \quad (17)$$

where $\Theta = (\boldsymbol{\pi}, \mathbf{A}, \mathbf{Y}, \beta)$

If an independent distribution is assumed for each parameter, then:

$$p(\Theta) = p(\pi)p(\mathbf{A})p(\mathbf{Y})p(\beta) \quad (18)$$

Taking into account the definitions in previous sections, the marginal likelihood in GTM-TT can be expressed, similarly to HMM (Beal, 2003), as:

$$p(\mathbf{X}) = \int p(\pi) \int p(\mathbf{A}) \int p(\mathbf{Y}) \int p(\beta) \sum_{\text{all paths}} p(\mathbf{Z}, \mathbf{X} | \pi, \mathbf{A}, \mathbf{Y}, \beta) d\beta d\mathbf{Y} d\mathbf{A} d\pi \quad (19)$$

Out of the many possible prior distributions to choose from, the conjugates of the distributions in Eqs.7-9 are a reasonable choice. In this way, a set of prior distributions is defined as follows:

$$\begin{aligned} p(\pi) &= \text{Dir}(\pi | \mathbf{v}) \\ p(\mathbf{A}) &= \prod_{k=1}^K \text{Dir}(\mathbf{a}_k | \boldsymbol{\lambda}) \\ p(\mathbf{Y}) &= [(2\pi)^K | \mathbf{C}|]^{-D/2} \prod_{d=1}^D \exp\left(-\frac{1}{2} \mathbf{y}_{(d)}^T \mathbf{C}^{-1} \mathbf{y}_{(d)}\right) \\ p(\beta) &= \Gamma(\beta | d_\beta, s_\beta) \end{aligned} \quad (20)$$

where \mathbf{a}_k is each of the column vectors of the matrix \mathbf{A} ; $\text{Dir}(\cdot)$ represents the Dirichlet distribution; and $\Gamma(\cdot)$ is the Gamma distribution. The prior over the parameter \mathbf{Y} defines the mapping from the hidden states to the data space as a GP, where $\mathbf{y}_{(d)}$ is each of the row vectors (centroids) of the matrix \mathbf{Y} and \mathbf{C} is the matrix of covariance functions. The vector \mathbf{v} of dimension K , the matrix $\boldsymbol{\lambda}$ of dimension $K \times K$, and the scalars ϵ and α (the parameters of the covariance functions) and the scalars d_β and s_β correspond to the hyperparameters of the model which are usually fixed a priori. Unfortunately, Eq.19 is analytically intractable. In the following section, we provide the details of its approximation using variational inference techniques.

A VARIATIONAL BAYESIAN APPROACH TO GTM THROUGH TIME

The Variational Bayesian EM Algorithm

Variational inference allows approximating the marginal log-likelihood through Jensen's inequality as follows:

$$\begin{aligned} \ln p(\mathbf{X}) &= \ln \int \sum_{\text{all paths}} p(\mathbf{Z}, \mathbf{X} | \Theta) p(\Theta) d\Theta \\ &\geq \int \sum_{\text{all paths}} q(\Theta, \mathbf{Z}) \ln \frac{p(\mathbf{Z}, \mathbf{X} | \Theta) p(\Theta)}{q(\Theta, \mathbf{Z})} d\Theta \\ &F(q(\Theta, \mathbf{Z})) \end{aligned} \quad (21)$$

The function $F(q(\Theta, \mathbf{Z}))$ is a lower bound such that its convergence guarantees the convergence of the marginal likelihood. The goal in variational inference is choosing a suitable form for the approximate density $q(\Theta, \mathbf{Z})$ in such a way that $F(q)$ can be readily evaluated and yet which is sufficiently flexible that the bound is reasonably tight. A reasonable approximation for $q(\Theta, \mathbf{Z})$ is based on the assumption that the hidden states \mathbf{Z} and the parameters Θ are independently distributed, i.e. $q(\Theta, \mathbf{Z}) = q(\Theta)q(\mathbf{Z})$. Thereby, a variational EM algorithm can be derived (Beal, 2003):

VBE-Step:

$$q(\mathbf{Z})^{(\text{new})} \leftarrow \arg \max_{q(\mathbf{Z})} F\left(q(\mathbf{Z})^{(\text{old})}, q(\Theta)\right) \quad (22)$$

VBM-Step:

$$q(\Theta)^{(\text{new})} \leftarrow \arg \max_{q(\Theta)} F\left(q(\mathbf{Z})^{(\text{new})}, q(\Theta)\right) \quad (23)$$

Variational Bayesian EM for GTM-TT

The VBE Step

The expression $q(\mathbf{Z})$ is estimated using, in eq.22, the complete-data likelihood in eq.13 and the prior probabilities in eq.20, so that:

$$\begin{aligned} \ln q(\mathbf{Z}) &= \left\langle \sum_{k=1}^K z_{k,1} \ln \pi_k \right\rangle_{q(\pi)} + \left\langle \sum_{n=2}^N \sum_{j=1}^K \sum_{k=1}^K z_{j,n-1} z_{k,n} \ln a_{jk} \right\rangle_{q(\mathbf{A})} \\ &+ \left\langle \frac{ND}{2} \ln \left(\frac{\beta}{2\pi} \right) \right\rangle_{q(\beta)} - \left\langle \frac{\beta}{2} \sum_{n=1}^N \sum_{k=1}^K z_{k,n} \|\mathbf{x}_n - \mathbf{y}_k\|^2 \right\rangle_{q(\mathbf{Y}, \beta)} \\ &- \ln \tilde{\mathcal{Z}}(\mathbf{X}) \end{aligned} \quad (24)$$

where $\ln \tilde{\mathcal{Z}}(\mathbf{X})$ is a normalization constant that depends on \mathbf{X} . Although it is expressed here in terms of the mean of the parameters of the model, this equation has a similar form to eq.13. Thus, a modified forward-backward procedure (Beal, 2003) can be used to solve it as follows:

$$\begin{aligned} \tilde{\alpha}(k, n) &= \frac{1}{\tilde{\zeta}(\mathbf{x}_n)} \left[\sum_{j=1}^K \tilde{\alpha}(j, n-1) \tilde{a}_{jk} \right] \tilde{p}(\mathbf{x}_n | z_n = k) \\ \text{with } \tilde{\alpha}(k, 1) &= \tilde{\pi}_k \end{aligned} \quad (25)$$

$$\begin{aligned} \tilde{\beta}(k, n) &= \sum_{j=1}^K \tilde{\beta}(j, n+1) \tilde{a}_{jk} \tilde{p}(\mathbf{x}_{n+1} | z_{n+1} = j) \\ \text{with } \tilde{\beta}(k, N) &= 1 \end{aligned} \quad (26)$$

where $\tilde{\pi}_k$ and \tilde{a}_{jk} are the estimated parameters; $\tilde{p}(\mathbf{x}_n \mid z_n = k)$ and $\tilde{p}(\mathbf{x}_{n+1} \mid z_{n+1} = j)$ are the emission probabilities calculated using the estimated parameters \mathbf{Y} and β ; and $\tilde{\zeta}(\mathbf{x}_n)$ is the normalization constant, related to the normalization constant of eq.24 by the expression $\prod_{n=1}^N \tilde{\zeta}(\mathbf{x}_n) = \tilde{\mathcal{Z}}(\mathbf{X})$.

The VBM Step

The variational distribution $q(\Theta)$ can be approximated to the product of the variational distribution of each one of the parameters if they are assumed to be independent and identically distributed. If so, $q(\Theta)$ can be expressed as: $q(\Theta) = q(\boldsymbol{\pi})q(\mathbf{A})q(\mathbf{Y})q(\beta)$, where natural choices of $q(\boldsymbol{\pi})$, $q(\mathbf{A})$, $q(\mathbf{Y})$ and $q(\beta)$ are similar distributions to the priors $p(\boldsymbol{\pi})$, $p(\mathbf{A})$, $p(\mathbf{Y})$ and $p(\beta)$, respectively. Thus:

$$\begin{aligned} q(\boldsymbol{\pi}) &= \text{Dir}(\boldsymbol{\pi} \mid \tilde{\mathbf{v}}) \\ q(\mathbf{A}) &= \prod_{k=1}^K \text{Dir}(\mathbf{a}_k \mid \tilde{\lambda}) \\ q(\mathbf{Y}) &= \prod_{d=1}^D \mathcal{N}(\mathbf{y}_{(d)} \mid \tilde{\mathbf{m}}_{(d)}, \tilde{\Sigma}) \\ q(\beta) &= \Gamma(\beta \mid \tilde{d}_\beta, \tilde{s}_\beta) \end{aligned} \quad (27)$$

Now, using eq.27 in eq.23, the following expressions for the variational parameters $\tilde{\mathbf{v}}$, $\tilde{\lambda}$, $\tilde{\Sigma}$, $\tilde{\mathbf{m}}$, \tilde{d}_β and \tilde{s}_β can be obtained:

$$\begin{aligned} \tilde{v}_k &= v_k + \langle z_{k,1} \rangle \\ \tilde{\lambda}_{j,k} &= \lambda_{j,k} + \sum_{n=2}^N \langle z_{j,n-1} z_{k,n} \rangle \\ \tilde{\Sigma} &= \left(\langle \beta \rangle \sum_{n=1}^N \mathbf{G}_n + \mathbf{C}^{-1} \right)^{-1} \\ \tilde{\mathbf{m}}_{(d)} &= \langle \beta \rangle \tilde{\Sigma} \sum_{n=1}^N x_{nd} \langle \mathbf{z}_n \rangle \\ \tilde{d}_\beta &= d_\beta + \frac{ND}{2} \\ \tilde{s}_\beta &= s_\beta + \frac{1}{2} \sum_{n=1}^N \sum_{k=1}^K \langle z_{k,n} \rangle \langle \| \mathbf{x}_n - \mathbf{y}_k \|^2 \rangle \end{aligned} \quad (28)$$

where \mathbf{G}_n is a diagonal matrix of size $K \times K$ with elements $\langle z_{k,n} \rangle$ in its diagonal. The moments in the previous equations are defined as:

$$\begin{aligned}
 \langle \beta \rangle &= \frac{\tilde{d}_\beta}{\tilde{s}_\beta} \\
 \langle \| \mathbf{x}_n - \mathbf{y}_k \| \rangle &= D\tilde{\Sigma}_{kk} + \| \mathbf{x}_n - \tilde{\mathbf{m}}_k \|^2 \\
 \langle z_{k,n} \rangle &= \frac{\tilde{\alpha}(k, n)\tilde{\beta}(k, n)}{\sum_{k=1}^K \tilde{\alpha}(k, N)} \\
 \langle z_{j,n-1} z_{k,n} \rangle &= \frac{\tilde{\alpha}(j, n-1)\tilde{a}_{jk}\tilde{p}(\mathbf{x}_n | z_n = k, \mathbf{Y}, \beta)\tilde{\beta}(k, n)}{\sum_{j'=1}^K \sum_{k'=1}^K \tilde{\alpha}(j', n-1)\tilde{a}_{j'k'}\tilde{p}(\mathbf{x}_n | z_n = k', \mathbf{Y}, \beta)\tilde{\beta}(k', n)}
 \end{aligned} \tag{29}$$

Note that expressions for $\langle z_{k,n} \rangle$ and $\langle z_{j,n-1} z_{k,n} \rangle$ are, in turn, the equivalent variational versions to $r_{k,n}$ and $\xi(n, j, k)$ in the standard GTM-TT.

Lower Bound Function

The lower bound function for the proposed VBGTM-TT is obtained by integrating out $F(q)$ (eq.21) through a similar procedure to the one described in (Beal, 2003), although, here, we must take into account the variational distributions of the parameters \mathbf{Y} and β , as follows:

$$\begin{aligned}
 F(q) &= \int \sum_{\text{all paths}} q(\boldsymbol{\pi})q(\mathbf{A})q(\mathbf{Y})q(\beta)q(\mathbf{Z}) \\
 &\times \ln \frac{p(\mathbf{Z}, \mathbf{X} | \boldsymbol{\pi}, \mathbf{A}, \mathbf{Y}, \beta)p(\boldsymbol{\pi})p(\mathbf{A})p(\mathbf{Y})p(\beta)}{q(\boldsymbol{\pi})q(\mathbf{A})q(\mathbf{Y})q(\beta)q(\mathbf{Z})} d\boldsymbol{\pi} d\mathbf{A} d\mathbf{Y} d\beta
 \end{aligned} \tag{30}$$

The solution for the lower bound is given by:

$$\begin{aligned}
 F(q) &= \ln \tilde{\mathcal{Z}}(\mathbf{X}) - D_{KL}[q(\boldsymbol{\pi}) || p(\boldsymbol{\pi})] - D_{KL}[q(\mathbf{A}) || p(\mathbf{A})] \\
 &\quad [-D_{KL} q(\mathbf{Y}) || p(\mathbf{Y})] - D_{KL}[q(\beta) || p(\beta)]
 \end{aligned} \tag{31}$$

where the operator $D_{KL}[q][p]$ is the Kullback-Leibler (KL) divergence between q and p . This equation implies that only the computation of the KL-divergence between the variational and the prior distribution for each parameter and the normalization constant are required to evaluate the lower bound function. Furthermore, the computation of the KL-divergence is straightforward because the distributions involved are all known.

EXPERIMENTS

Experimental Design

The goal of the set of experiments presented in this section is the assessment of the performance of the proposed VBGTM-TT model, using MTS contaminated with increasing levels of Gaussian noise, and for different numbers of states.

The models used in all the experiments were initialized in the same way to allow straightforward comparison. The matrix centroids of the Gaussian generators \mathbf{Y} and the inverse of the variance β were set through PCA-based initialization. The hyperparameter s_β was set to d_β/β and d_β was initialized to a small value close to 0. For each set of experiments, several values of α were tried, finally settling for a value of 0.1.

The following three datasets were selected for the experiments:

- *Spiral3d_data*: A simple three-dimensional artificial dataset consisting of 500 data points that was artificially generated using the equation of a spiral 3D contaminated with Gaussian noise, as follows:

$$\mathbf{X} = \begin{bmatrix} x_1 = t \cos t + \sigma_0 \\ x_2 = t \sin t + \sigma_0 \\ x_3 = t \end{bmatrix}$$

where $t = 3\pi(1 + n/250)$, $1 \leq n \leq 500$ and σ_0 is an added Gaussian noise.

- *Shuttle data*: A real dataset consisting of 6 time series and 1000 data points.
- *Shuttle_smooth_data*: a smoothed, low pass-filtered version of *Shuttle_data* (See Figure 1)

Figure 1. Plot of the Shuttle_smooth_data dataset. These data consisting of 1000 data points obtained from various inertial sensors from Space Shuttle mission STS-57. Available from URL: www.cs.ucr.edu/~eamonn. Each of the six series is represented using a different line style.

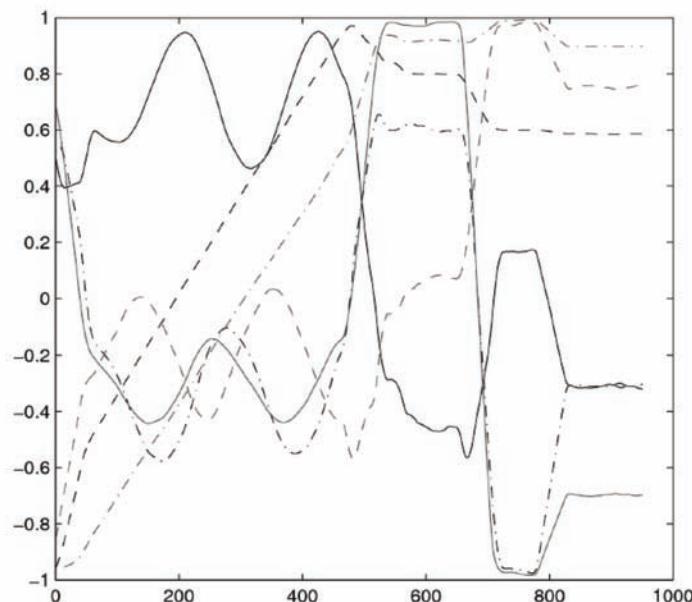
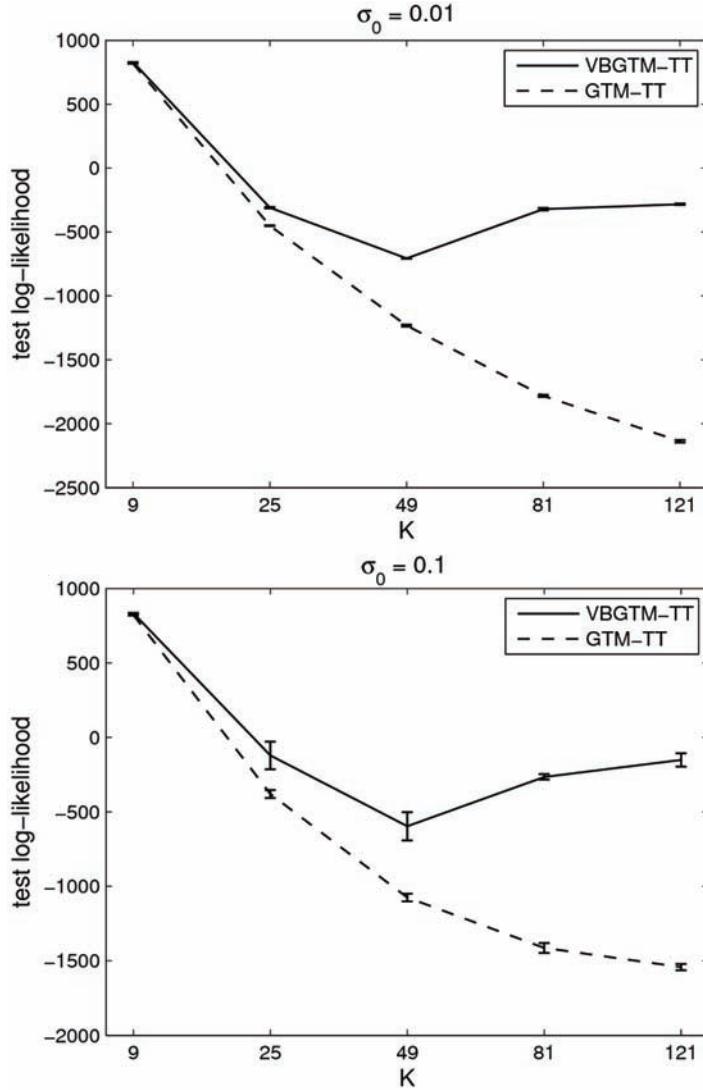


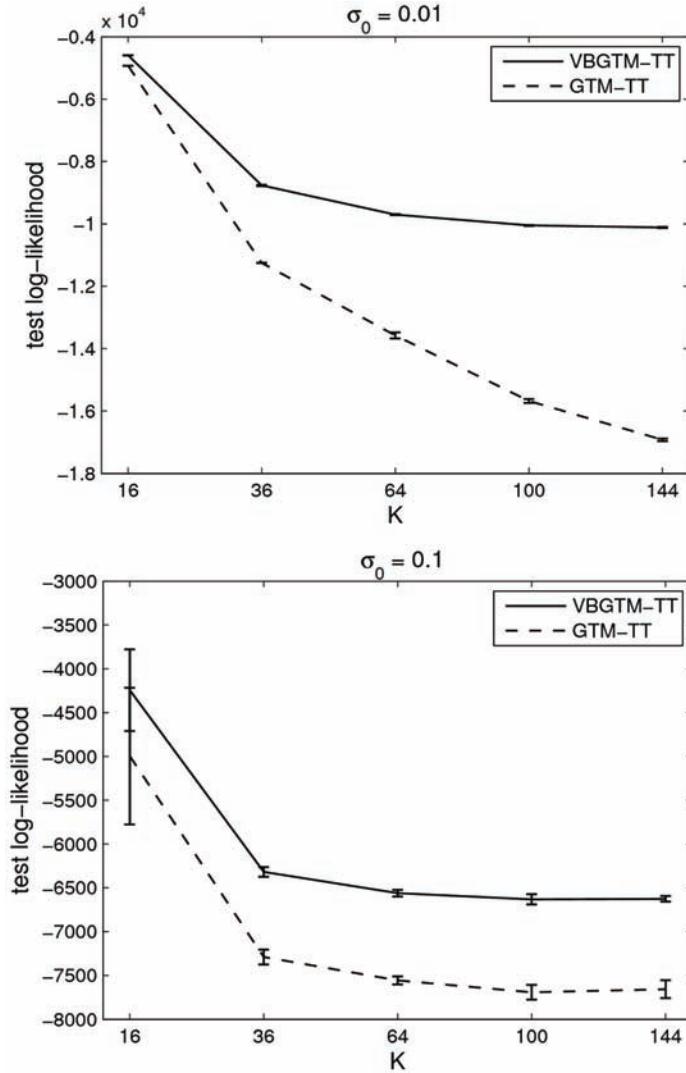
Figure 2. Average test log-likelihood for the proposed VBGTM-TT (solid line) and standard. GTM-TT with GP prior (dashed line) over 10 runs of the algorithms, for different numbers of states and different levels of added Gaussian noise using the *Spiral3d_data* dataset. Top: 0.01 standard deviation, bottom: 0.10. The vertical bars span from the average minus one standard deviation of the 10 runs, to the average plus one standard deviation.



Comparative Assessment of the Robustness of VBGTM-TT in the Presence of Noise

The test log-likelihood was used to assess the performance of VBGTM-TT and GTM-TT with GP prior models. Several versions of *Spiral3d_data* and *Shuttle_smooth_data*, contaminated with two levels of Gaussian noise, were generated to train the models. Different numbers of hidden states K were used for each training dataset. More precisely, 10 noisy datasets were generated for each noise level and each

Figure 3. Average test log-likelihood for the proposed VBGTM-TT (solid line) and standard. GTM-TT with GP prior (dashed line) over 10 runs of the algorithms, for different numbers of states and different levels of added Gaussian noise using the *Shuttle_smooth_data* dataset. Top: 0.01 standard deviation, bottom: 0.10. The vertical bars span from the average minus one standard deviation of the 10 runs, to the average plus one standard deviation.



value of K . In all experiments, the original *Spiral3d_data* and *Shuttle_smooth_data* datasets without noise were used for testing the models. Results are displayed, in turn, in Figure 2 for *Spiral3d_data*, and in Figure 3 for *Shuttle_smooth_data*.

The results that can be observed in these figures show that the proposed VBGTM-TT outperformed GTM-TT with GP prior in all experiments. The performance of the VBGTM-TT was significantly better in experiments with a large number of hidden states. The influence of the contamination by Gaussian noise does not seem to affect the performance of the VBGTM-TT significantly. However, the performance gap between both methods seems to reduce when the level of the noise is higher (standard deviation 0.1).

On the Influence of the VB Approximation for GTM-TT in the Visualization of the Multivariate Time Series

As already mentioned, one of the most interesting capabilities of Variational Bayesian GTM-TT is that of providing a visualization of the MTS and their evolution over the model states. We would expect the state representation resulting from the training of the standard GTM-TT with GP prior to be more affected by the presence of noise than that of its variational counterpart. This would entail a profusion of small states reflecting the noise in the former, paired with a jittery trajectory through them, while a more parsimonious state distribution and a more stable trajectory through the state space would be expected from the training results of the latter.

In (Olier & Vellido, 2008), the GTM-TT standard model was used to visualize subsequences in the *Shuttle_data* dataset. Four subsequences, named A, C, D and E (see Figure 4), were identified as little variability periods. Furthermore, a fifth subsequence, named B, was identified as a high variability period.

These subsequences are now highlighted in the membership maps resulting from GTM-TT with GP prior and VBGTM-TT, which are displayed in Figures 5 and 6, respectively. The visualization of the evolution of the time series through the map of states defined by VBGTM-TT is much more parsimonious than that defined by the GTM-TT with GP prior, with far less states required to describe the little

Figure 4. Four periods of relatively little variability in Shuttle_data, separated by sudden transitions, are singled out as A, C, D and E. A fifth period B of higher variability, also delimited by sudden transitions, spans from the end of A to the beginning of C.

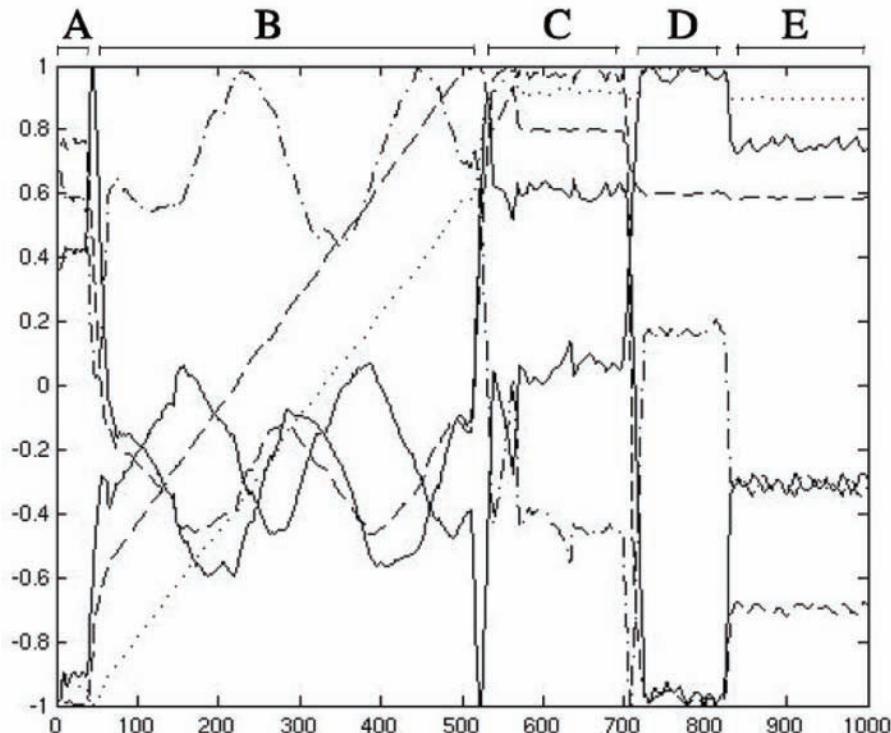


Figure 5. Membership map representing the *Shuttle_data* dataset using the standard GTM-TT with GP prior method. The hidden states are connected by solid lines representing the transitions between them. The previously detected subsequences A, B, C, D and E are highlighted by ellipses of dashed lines.

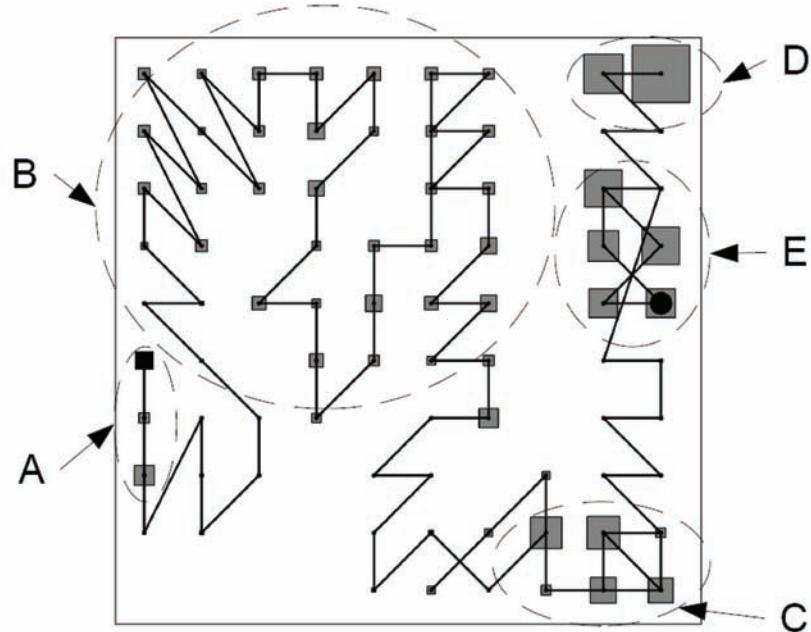
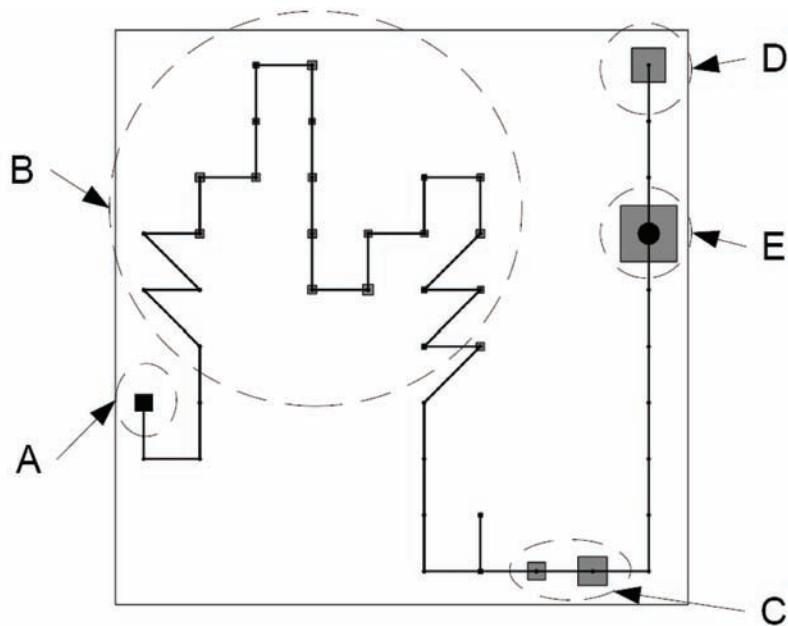


Figure 6. Membership map representing the *Shuttle_data* dataset using the VBGTM-TT mode, using the same representation as in previous figure.



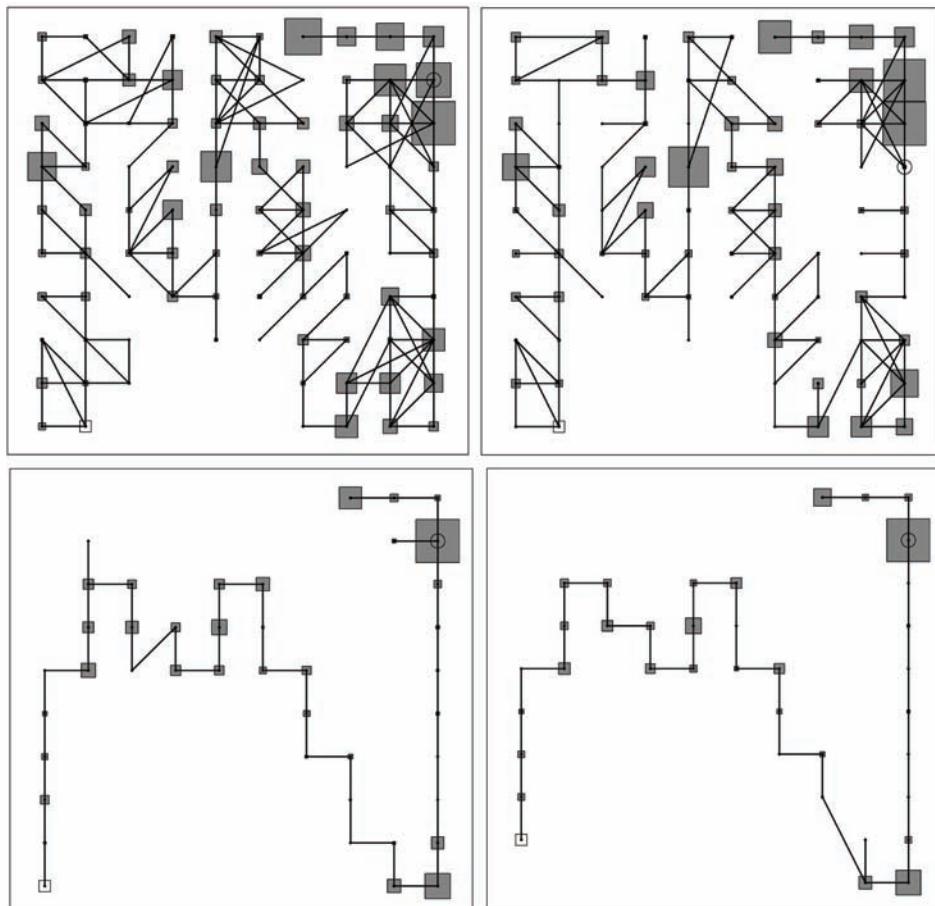
variability periods, as expected. VBGTM-TT is further shown to be less sensible to the added uninformative noise than the GTM-TT with GP prior, as illustrated by the far more jittery transitions between states of the latter.

Further experiments were carried out, this time using the *Shuttle_smooth_data* dataset, which was contaminated with Gaussian noise with standard deviation of 0.2. Two datasets were generated, one for training and another for test. The resulting membership maps for GTM-TT with GP prior and for VBGTM-TT are displayed in Figure 7 and confirm the previous results. Once again, VBGTM-TT is shown to be less sensible to the presence of noise. Furthermore, the visualization of the test dataset is almost identical to that of the training dataset in VBGTM-TT.

CONCLUSION

In this chapter, we have extended the variational Bayesian formulation proposed for GTM in (Olier & Vellido, 2008) to the unsupervised analysis, through clustering and visualization, of MTS. With it, we

Figure 7. Membership maps representing the Shuttle_smooth_data dataset. Maps on the top row display the results for GTM-TT with GP prior and on the bottom row, for VBGTM-TT. Maps on the left-hand side are for the training datasets and on the right-hand side, for the test datasets.



aimed to provide a principled solution to an important limitation of the standard GTM formulation for time series (the GTM-TT): its lack of regularization scheme, with the corresponding risk of overfitting, which is commonplace in real problems.

REFERENCES

- Abrahamsen, P. (1997). *A review of Gaussian random fields and correlation functions* (Tech. Rep. 917). Norwegian Computing Center, Oslo, Norway.
- Baum, L., & Egon, J. (1967). An inequality with applications to statistical estimation for probabilistic functions for a Markov process and to a model for ecology. *Bulletin of the American Meteorological Society*, 73, 360–363. doi:10.1090/S0002-9904-1967-11751-8
- Beal, M. (2003). *Variational algorithms for approximate Bayesian inference*. Unpublished doctoral dissertation, The Gatsby Computational Neuroscience Unit, University College London.
- Bishop, C., Hinton, G., & Strachan, I. (1997). GTM through time. In *Proceedings of the IEEE Fifth International Conference on Artificial Neural Networks* (pp.111-116).
- Bishop, C., Svensén, M., & Williams, C. (1998a). GTM: The generative topographic mapping. *Neural Computation*, 10(1), 215–234. doi:10.1162/089976698300017953
- Bishop, C., Svensén, M., & Williams, C. (1998b). Developments of the generative topographic mapping. *Neurocomputing*, 21(1-3), 203–224. doi:10.1016/S0925-2312(98)00043-5
- Bullen, R. J., Cornford, D., & Nabney, I. T. (2003). Outlier detection in scatterometer data: Neural network approaches. *Neural Networks*, 16(3-4), 419–426. doi:10.1016/S0893-6080(03)00013-3
- Carreira-Perpiñan, M. A. (2000). Reconstruction of sequential data with probabilistic models and continuity constraints. In S. Solla, T. Leen, & K. R. Muller (Eds.), *Advances in neural information processing systems 12* (pp. 414-420). San Francisco, CA: Morgan Kauffman.
- Chappell, G., & Taylor, J. (1993). The temporal Kohonen map. *Neural Networks*, 6, 441–445. doi:10.1016/0893-6080(93)90011-K
- Girolami, M. (2002). Latent variable models for the topographic organisation of discrete and strictly positive data. *Neurocomputing*, 48, 185–198. doi:10.1016/S0925-2312(01)00659-2
- Kabán, A., & Girolami, M. (2002). A dynamic probabilistic model to visualise topic evolution in text streams. *Journal of Intelligent Information Systems*, 18(2/3), 107–125. doi:10.1023/A:1013673310093
- Keogh, E., & Lin, J. (2005). Clustering of time series subsequences is meaningless: Implications for previous and future research. *Knowledge and Information Systems*, 8(2), 154–177. doi:10.1007/s10115-004-0172-7
- Kohonen, T. (2001). *Self-organizing maps* (3rd ed.). Berlin, Germany: Springer-Verlag.
- Kostiainen, T., & Lampinen, J. (2002). On the generative probability density model in the self-organizing map. *Neurocomputing*, 48(1-4), 217–228. doi:10.1016/S0925-2312(01)00649-X

- Liao, T. W. (2005). Clustering of time series data - a survey. *Pattern Recognition*, 38(11), 1857–1874. doi:10.1016/j.patcog.2005.01.025
- MacKay, D. J. C. (1997). *Ensemble learning for Hidden Markov Models* (Tech. Rep.). Cavendish Laboratory, University of Cambridge, U.K.
- Olier, I., & Vellido, A. (2008). Advances in clustering and visualization of time series using GTM through time. *Neural Networks*, 21(7), 904–913. doi:10.1016/j.neunet.2008.05.013
- Olier, I., & Vellido, A. (2008). Variational Bayesian generative topographic mapping. *Journal of Mathematical Modelling and Algorithms*, 7(4), 371–387. doi:10.1007/s10852-008-9088-7
- Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77, 257–285. doi:10.1109/5.18626
- Simon, G., Lee, J., & Verleysen, M. (2006). Unfolding preprocessing for meaningful time series clustering. *Neural Networks*, 19(6-7), 877–888. doi:10.1016/j.neunet.2006.05.020
- Stolcke, A., & Omohundro, S. (1993). Hidden Markov model induction by Bayesian model merging. In S. J. Hanson, J. D. Cowan, & C. L. Giles (Eds.), *Advances in neural information processing systems*, vol. 5 (pp. 11-18). San Francisco, CA: Morgan Kauffmann.
- Strickert, M., & Hammer, B. (2005). Merge SOM for temporal data. *Neurocomputing*, 64, 39–71. doi:10.1016/j.neucom.2004.11.014
- Tiňo, P., Farkaš, I., & van Mourik, J. (2006). Dynamics and topographic organization of recursive self-organizing maps. *Neural Computation*, 18, 2529–2567. doi:10.1162/neco.2006.18.10.2529
- Tiňo, P., & Nabney, I. (2002). Hierarchical GTM: Constructing localized nonlinear projection manifolds in a principled way. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), 639–656. doi:10.1109/34.1000238
- Vellido, A. (2006). Missing data imputation through GTM as a mixture of t -distributions. *Neural Networks*, 19(10), 1624–1635. doi:10.1016/j.neunet.2005.11.003
- Vellido, A., El-Deredy, W., & Lisboa, P. J. G. (2003). Selective smoothing of the generative topographic mapping. *IEEE Transactions on Neural Networks*, 14(4), 847–852. doi:10.1109/TNN.2003.813834
- Vellido, A., & Lisboa, P. J. G. (2006). Handling outliers in brain tumour MRS data analysis through robust topographic mapping. *Computers in Biology and Medicine*, 36(10), 1049–1063. doi:10.1016/j.combiomed.2005.09.004
- Voegtlin, T. (2002). Recursive self-organizing maps. *Neural Networks*, 15(8-9), 979–991. doi:10.1016/S0893-6080(02)00072-2
- Yin, H., & Allinson, N. (2001). Self-organizing mixture networks for probability density estimation. *IEEE Transactions on Neural Networks*, 12, 405–411. doi:10.1109/72.914534

KEY TERMS AND DEFINITIONS

Clustering: The process of assigning individual data items into groups (called clusters) so that items from the same cluster are more similar to each other than items from different clusters. Often similarity is assessed according to a distance measure.

Data Visualization: visual representation of data, aiming to convey as much information as possible through visual processes

Generative Topographic Mapping: A nonlinear model for dimensionality reduction. It belongs to the manifold learning family and performs simultaneous data clustering and visualization.

Latent Variable Model: a statistical or machine learning model that relates a set of variables (observable variables, residing in observed data space) to set of latent variables

Nonlinear Dimensionality Reduction: the process of representing high-dimensional data in lower-dimensional spaces through mapping, projection, feature selection and other methods

Multivariate Time Series: In statistics, signal processing, and many other fields, a multivariate time series is a set of sequences of data points, measured typically at successive times, spaced at (often uniform) time intervals

Variational Bayesian Methods: A family of techniques for approximating intractable integrals arising in Bayesian statistics and machine learning. They can be used to lower bound the marginal likelihood of several models with a view to performing model selection, and often provide an analytical approximation to the parameter posterior probability which is useful for prediction

Chapter 9

Locally Recurrent Neural Networks and Their Applications

Todor D. Ganchev
University of Patras, Greece

ABSTRACT

In this chapter we review various computational models of locally recurrent neurons and deliberate the architecture of some archetypal locally recurrent neural networks (LRNNs) that are based on them. Generalizations of these structures are discussed as well. Furthermore, we point at a number of real-world applications of LRNNs that have been reported in past and recent publications. These applications involve classification or prediction of temporal sequences, discovering and modeling of spatial and temporal correlations, process identification and control, etc. Validation experiments reported in these developments provide evidence that locally recurrent architectures are capable of identifying and exploiting temporal and spatial correlations (i.e., the context in which events occur), which is the main reason for their advantageous performance when compared with the one of their non-recurrent counterparts or other reasonable machine learning techniques.

INTRODUCTION

After the introduction of the simplified computational model of a neuron in the early 1940's (McCulloch & Pitts, 1943), various advanced models of neurons and derivative neural network structures were developed. Nowadays, i.e. sixty-five years later, there is a huge number of different neurons and immense number of neural networks, which differ by their building blocks, linkage, and structure. This abundance of architectures reflects the wide-spread use of neural networks on real-life problems, and the diversity of these problems attests to the inherent flexibility and enormous potential of neural networks.

DOI: 10.4018/978-1-60566-766-9.ch009

In the present chapter we focus our attention to one class of dynamic neural networks, namely the locally recurrent architectures that were introduced in the beginning of the 1990's, and that regained significant attention in the past few years.

The emergence of locally recurrent architectures contributed to significant advance in the theory and practice of neural networks. In the years following their introduction, numerous derivative architectures flourished, and the locally recurrent neurons become important building blocks in many original neural network designs, developed for the needs of particular real-world applications. In these applications, LRNNs were proved successful in dealing with spatial correlations, time sequences and structured data. Moreover, the engagement of LRNNs in real-world applications required with improved training strategies and variety of new architectures. The considerable research interest to LRNNs led to significant advance of the gradient-based learning techniques that had to be adapted for the needs of temporal sequences processing. Furthermore, numerous new training methods based on genetic algorithms, particle swarm optimization (PSO), and other evolutionary techniques appeared, diversifying the nomenclature of training methods and improving the applicability of LRNNs on specific domain-oriented tasks or in general.

Before proceeding with the main exposition, we would like to define few terms that we use through the text to avoid potential ambiguity:

First of all, we name *dynamic neural networks* these networks that incorporate dynamic synaptic or feedback weights among some or all of their neurons. These networks are capable of expressing dynamic behaviors.

Recurrent neural networks we label those architectures which incorporate feedback connections among the layers of the network, or those that do not have straightforward layered input-output architecture but instead the inputs flow forth and back among the nodes of the network.

Feedforward architectures that incorporate a layer of recurrent neurons, which possess feedbacks only from neurons belonging to the same layer, are referred to as *locally connected* recurrent neural networks. The locally connected architectures are also referred to as *non-local recurrent* structures.

Next, here we use the term *locally recurrent neural network* for any network that possesses neurons with local feedback connections. On the level of individual neurons, local recurrence is identified with the availability of one or more feedbacks that encompass one or more elements in the structure of a neuron. The following section discusses this matter in details.

Furthermore, on the level of neural networks (we assume multilayer neural networks) the local recurrent neurons might be organized in a feedforward, non-local recurrent, or global recurrent architectures, depending on the implementation of the network linkage. For further reference to terminology we direct the interested reader to the work of Tsoi and Back (Tsoi and Back, 1997), which offers a comprehensive overview of terms and architectures related to recurrent neural networks.

In the following sections we overview various neural network architectures that are based on locally recurrent neurons. We place the accent on detailing the structure of these neural networks and, therefore, the numerous training methods developed for these architectures are not covered in details. The interested reader shall follow the comprehensive references provided in order to obtain more information about the training methods used, and their performance on real-world problems.

In conclusion of this brief introduction, we would like to note that here we do not aim at exhaustive enumeration of all existing applications of LRNNs, nor do we intend to advocate or validate any LRNN-based solution against alternatives. Instead, we intend to list some examples of successful application of LRNNs on real-world problems, which illustrate the diversity and flexibility of these architectures, or which offer a perspective for prospective future applications. Thus, in the following sections, our

interest falls mostly on applications that are related to processing of structured data, or on the various signal processing applications that require dealing with spatial or temporal correlations. Some indicative examples among others are: active noise cancelation (Jayawardhana et al., 2002), speech recognition (Bengio et al., 1992; Kasper et al, 1995), speaker recognition (Ganchev et al., 2007), emotion detection (Ganchev et al., 2008), face recognition (Psarrou et al., 1995), sound synthesis (Uncini 2002), wind speed estimation (Barbounis et al., 2006; Barbounis & Theocharis, 2007a; Barbounis & Theocharis, 2007b), system identification (Back & Tsoi, 1992), various prediction problems (Mandic & Chambers, 2001), control applications (Ku & Lee, 1995; Cadini et al., 2008), etc.

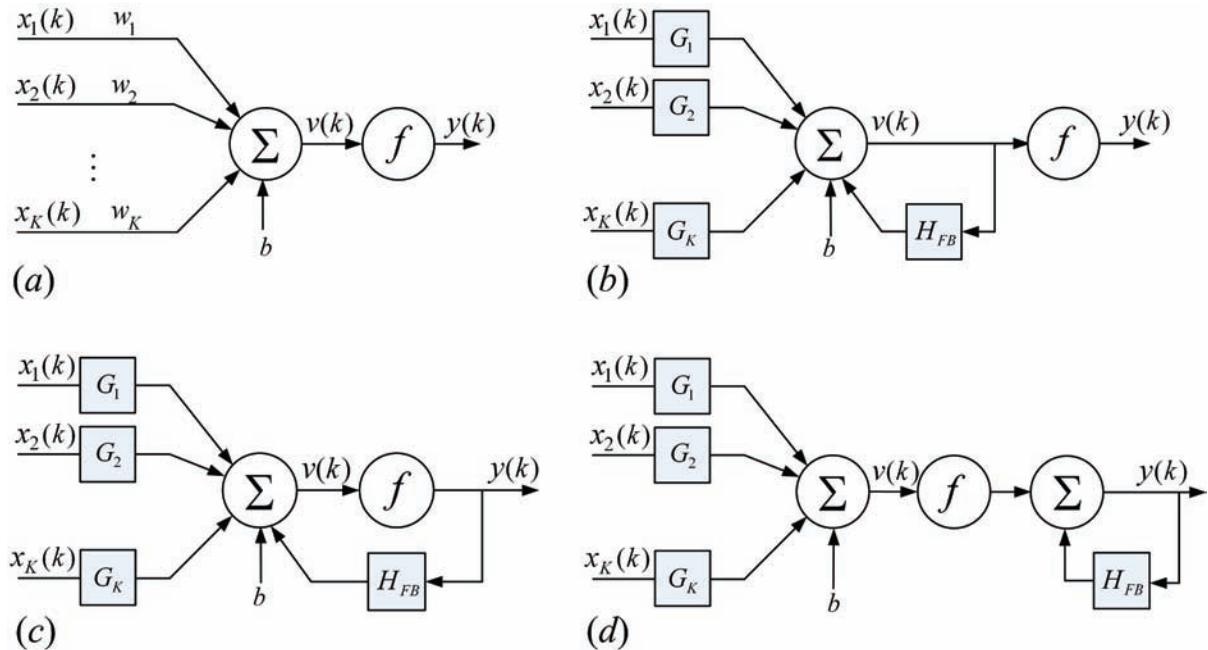
COMPUTATIONAL MODELS OF LOCALLY RECURRENT NEURONS

In this section, we briefly review various locally recurrent structures that have been studied in the literature. Computational models of neurons, based on these structures, serve as the building blocks in the design of LRNNs. Typical neural network paradigms are discussed in the following sections.

In Figure 1 we illustrate the general internal structure of four archetypal computational models of a neuron. Here, the arrows that link the building blocks of the neuron (or that link neurons in the following figures) serve to emphasize the data flow, i.e. they do not indicate for the presence of (synaptic) weights.

Specifically, Figure 1 (a) shows the computational model of a static feedforward neuron, whose synaptic weights w_i are constants. According to the formulation in (Lapedes & Faber, 1987) the output, $y(k)$, of such a neuron can be expressed as:

Figure 1. Computational model of a neuron: (a) static neuron, (b) general form of a locally recurrent structure with activation feedback, (c) general form of a locally recurrent structure with output feedback, and (d) general form of a locally recurrent structure with autoregressive feedback.



$$\begin{aligned} y(k) &= f(v(k)) \\ v(k) &= \sum_{i=1}^K w_i x_i(k) + b , \end{aligned} \quad (1)$$

where b is the bias, k is the discrete time, and $x_i(k)$ with $i=1,2,\dots,K$ is the input for synapse i at the time instant k . This architecture is essentially identical to the computational model of a neuron that was proposed by McCulloch and Pitts (1943).

However, since the static neuron is a very raw approximation of the biological neuron, more advanced (and reportedly with closer biological plausibility – refer to Uchiyama et al., (1989); Natschläger et al., (2000); Natschläger et al., (2001); etc) computational models were developed. In these models the static synaptic weights are replaced by dynamic ones, but more importantly, various types of local feedbacks inside the neuron were introduced. These feedbacks encompass one or more elements in the structure of the artificial neuron and provide the means for incorporating memory about past states. Specifically, depending on the feedback type, four major types of locally recurrent architectures can be identified. These, as defined in (Tsoi & Back, 1994; Campolucci et al., 1999), are:

- (i) these with feedback from the synapse output,
- (ii) these with activation feedback, i.e. feedback before the activation function,
- (iii) these with output feedback, i.e. feedback from the neuron output, and
- (iv) these with autoregressive filter after the output of the activation function.

The first architecture, i.e. neurons with synaptic feedback, was introduced by Back and Tsoi (Back & Tsoi, 1991), who replaced the static synaptic weights with infinite impulse response (IIR) filters. The resulting structure implements dynamic synaptic weights which allow modelling temporal dependences in sequences. In the case when the transfer function, G_i , of the dynamic synapses is implemented with a linear transfer function with poles and zeros, the output of the neuron can be written (Tsoi & Back, 1994) as:

$$\begin{aligned} y(k) &= f(v(k)) \\ v(k) &= \sum_{i=1}^K G_i(z^{-1}) x_i(k) + b , \end{aligned} \quad (2)$$

where the linear transfer function, G_i , has the general form:

$$G(z^{-1}) = \frac{\sum_{j=0}^{m_z} b_j z^{-j}}{\sum_{j=0}^{n_p} a_j z^{-j}} . \quad (3)$$

Here m_z and n_p are the number of zeroes and the number of poles in the linear transfer function, a_j with $j=0,1,2,\dots$, n_p are the poles, b_j with $j=0,1,2,\dots$, m_z are the zeroes and z^{-j} are the delay elements defined as $z^{-j} x(k) \equiv x(k-j)$.

As it was noted in (Campolucci et al., 1999), the synaptic feedback architecture can be viewed as a

special case of the locally recurrent activation feedback architecture, shown in Figure 1 (b). As the figure presents, the locally recurrent activation feedback architecture implements a feedback from the output of the summation unit. In this way, at the input of the summation unit of the neuron acts the combination of the processed via G_i inputs, $x_i(k)$, and the processed via H_{FB} summation output, $v(k-1)$, obtained at the previous time instant. This structure has been defined in (Tsoi & Back, 1994), where the authors review and summarize the ideas of various locally recurrent structures, such as the: FIR and IIR synapses (Back & Tsoi, 1991), generalized neuron of Frasconi-Gori-Soda (Frasconi et al, 1992), neurons with gamma-memory (de Vries & Principe, 1992; Principe et al., 1993), etc.

Specifically, the local recurrent activation feedback structure, in its general form, is described by the following formulation:

$$\begin{aligned} y(k) &= f(v(k)) \\ v(k) &= \sum_{i=1}^K G_i(z^{-1})x_i(k) + H_{FB}(z^{-1})v(k-1) + b, \end{aligned} \quad (4)$$

where $H_{FB}(z^{-j})$, in the general case, may possess both zeros and poles:

$$H_{FB}(z^{-1}) = \frac{\sum_{j=0}^{g_z} b_j z^{-j}}{\sum_{j=0}^{l_p} a_j z^{-j}}. \quad (5)$$

Here, g_z and l_p are the number of zeroes and the number of poles in the linear transfer function, H_{FB} .

As pointed out in (Tsoi & Back, 1994; Campolucci et al., 1999) the activation feedback architecture can be viewed as a particular case of the IIR synaptic feedback, where all synaptic transfer functions, G_i , have a common denominator. In this way, each synapse can be considered as an individual local activation feedback structure, where the local activations of various synapses are summed before their entry to the nonlinear element, f .

Learning algorithms for activation feedback neurons were studied in (Gori, 1989; Gori et al., 1989; Gori & Soda, 1990; Frasconi et al., 1992; and elsewhere).

In the local output feedback architecture, shown in Figure 1 (c), the feedback loop embraces the non-linear activation function. In this way at the input of the summation unit together with the inputs, $x_i(k)$, processed by the transfer functions, G_i , act also the delayed outputs of the neuron processed by the transfer function, H_{FB} . Therefore, the output of a neuron with local output feedback can be expressed as:

$$\begin{aligned} y(k) &= f(v(k)) \\ v(k) &= \sum_{i=1}^K G_i(z^{-1})x_i(k) + H_{FB}(z^{-1})y(k-1) + b . \end{aligned} \quad (6)$$

In the general case, the transfer functions G_i and H_{FB} may possess both zeros and poles, as formulated by equations (3) and (5), respectively.

In the locally recurrent structure with autoregressive feedback, shown in its general form in Figure 1(d), autoregressive post-processing of the output of the non-linear function f is performed. This structure was originally proposed in (Mozer, 1989) and developed further in (Leighton & Conrath, 1991). In its general form, the output of a neuron with autoregressive feedback can be written as:

$$\begin{aligned} y(k) &= f(v(k)) + H_{FB}(z^{-1})y(k-1) \\ v(k) &= \sum_{i=1}^K G_i(z^{-1})x_i(k) + b . \end{aligned} \quad (7)$$

Again, here the transfer functions G_i and H_{FB} may possess both zeros and poles.

Let's go back to Figure 1 (b), where the general form of locally recurrent structure with activation feedback is shown. The transfer functions G_i and H_{FB} can be implemented in various ways, which would lead to different variants of the general structure. For instance, the FIR and IIR synapses (Back & Tsoi, 1991) are obtained by defining G_i as either FIR or IIR structure, respectively, and discarding the feedback loop with transfer function H_{FB} . In addition, the IIR synapse can be obtained by defining H_{FB} to act as the common denominator for all synaptic weights G_i , defined as FIR filters. Furthermore, the generalized neuron of Frasconi-Gori-Soda can be obtained from the general form of locally recurrent structure with output feedback by replacing the dynamic synaptic weights of the inputs with static ones.

Finally, it is worth to mention the study of Mozer (1993), who investigated various implementations of tapped delay line memories, which are common building blocks in the locally recurrent architectures. His study offers insights about the behaviour of different delay lines, and shows some interesting aspects of the LRNNs design that were found to play important role in the practical implementation of these structures.

IMPLEMENTATIONS OF LOCALLY RECURRENT NEURAL NETWORKS

Various categorizations of neural network architectures can be devised. Here we consider only these architectures that involve locally recurrent neurons. According to the nomenclature description provided in (Tsoi & Back, 1997) there are two basic types of locally recurrent neural networks: (i) locally recurrent globally feedforward, and (ii) locally recurrent globally recurrent. In the former, the neural network architecture is globally feedforward, i.e. there are no feedbacks among different layers, and includes locally recurrent neurons either in its input, hidden or output layers. On the other hand, the locally recurrent globally recurrent neural networks possess feedbacks among the layers. Very often these feedbacks are from the output to the input layer, but there are numerous neural network paradigms where the feedbacks are also from the hidden layers to the input or from the output layer to the hidden layers.

Two other related categories are the non-locally recurrent globally feedforward and the non-locally recurrent globally recurrent neural networks. These differ from their locally recurrent counterparts only by the availability of inter-neuron connections among the locally recurrent neurons belonging to the same layer. Furthermore, it is possible that in the same layer of the neural network, some of the recurrent neurons are linked with each other (forming a specific local neighbourhood, also referred to as a *block*) but others are linked only to the previous and next layers, and thus possess only local feedbacks. One example for such an architecture is the partially connected locally recurrent probabilistic neural network

(Ganchev et al., 2008), which in the hidden recurrent layer may possess both neurons with non-locally recurrent connections and neurons with locally recurrent feedbacks. The non-locally recurrent globally feedforward architecture belongs to the large group of locally connected recurrent neural networks (Chan & Young, 1995; Elizondo & Fiesler, 1997; Thomas, 2000; Grassi & Cafagna, 2003; etc).

The locally recurrent neurons briefly outlined in the previous section, as well as an assortment of similar or derivative structures, serve as the basis in the design of locally recurrent neural networks. There exist a huge number of locally recurrent neural networks that already have been investigated in the literature (Ku et al., 1992; Ciocoiu, 1996; Zemouri et al., 2003; Barbounis et al., 2006; Barbounis & Theocharis, 2007a; Ganchev et al., 2008; etc) and have been proved valuable on specific tasks. In the following subsections, we review these architectures and illustrate few typical members of each category.

Multilayer Perceptron Neural Networks with FIR and IIR Synapses

Multilayer perceptron (MLP) locally recurrent globally feedforward neural networks can be viewed as natural extension of the static MLP, where the synaptic weights of some or all neurons are replaced by linear or non-linear dynamic structures. As it was discussed already, examples of dynamic structures, such as FIR, IIR, gamma-memory etc synapses, have been studied widely in the literature (Back & Tsoi, 1991; de Vries & Principe, 1992; etc). Further extensions of the locally recurrent MLP NN involve the incorporation of activation or output local feedbacks, as well as the integration of autoregressive filters in the output of the locally recurrent neurons (Gori et al., 1989; Frasconi et al., 1992; etc).

Just like their static counterparts the locally recurrent MLP NNs can possess two or more layers of neurons. Depending on the desired functionality and the specifics of the target application these MLP NNs may incorporate neurons with dynamic synaptic weights or local feedbacks in the input, hidden or the output layers. However, due to practical considerations related to complexity of training, availability of a limited amount of training data, etc, MLP NNs with two or three layers, one or two of them containing locally recurrent neurons, are the most typical choice in real-world applications.

Nowadays, there is a variety of gradient-based and evolutionary algorithms for training locally recurrent MLP NNs, which have been reported to offer specific advantages in different application domains. Specifically, Back and Tsoi (1991) proposed gradient-based learning strategy for training locally recurrent MLP NNs with IIR-synapses (IIR-MLP NN). A comprehensive study of several MLP-based LRNNs is available in (Campolucci et al., 1999). They introduced a unifying framework for the gradient-based calculation techniques, referred to as causal recursive back-propagation, which generalizes several earlier methods. In (McDonnell & Waagen, 1994), evolutionary strategies for training IIR-MLP NNs were studied. The hybrid stochastic optimization approach proposed in that work was applied in determining both the model order and the coefficients of recurrent perceptron time-series models.

MLP NNs based on the FIR and IIR synapses and/or possessing feedbacks were found advantageous in various applications, when compared to their static counterparts.

Applications of the FIR and IIR-based MLP NNs

Bengio et al. (1992) studied a three layer MLP-based NN, which incorporates the Frasconi-Gori-Soda type of locally recurrent neurons in the hidden layer. The local recurrence and the forgetting behaviour of this structure, with respect to previous context, were found useful on phoneme recognition task.

Back and Tsoi (1992) illustrated the usefulness of IIR-MLP NN on non-linear system identification tasks, such as Hammerstein and Wiener system identification. Moreover, the authors illustrated how a number of IIR-MLP structures can be used for representing the Uryson nonlinear system. In conclusion of that work the authors reported that the IIR-MLP-based NNs offer an effective way of modelling block-oriented nonlinear systems.

Campolucci et al. (1996) investigated the performance of a buffered MLP, FIR-MLP and IIR-MLP NNs with improved training on the task of identifying a baseband equivalent pulse amplitude modulation (PAM) transmission system in presence of nonlinearity. Based on the experimental results the authors concluded that the slightly higher computational complexity of the proposed training algorithm, which is approximation of the Back and Tsoi algorithm, is justified by better stability and higher speed of convergence.

Cannas et al. (1998a) offer performance analysis of a three-layer LRNN that incorporates IIR neurons in the hidden layer. They performed an analysis of the capabilities of FIR- and IIR-based NNs on forecasting chaotic time series and evaluation of their ability to model and simulate the behaviour of autonomous non-linear circuits. The authors reported that the evaluated locally recurrent networks are able to model the Chua's circuits (Chua & Lin, 1990; Chua, 1993; Chua et al., 1996) considered in their study.

Later on, Cannas et al. (1998b) utilized an IIR-MLP NN in a control scheme, which aims at maintaining a good quality of voltage in a power distribution system. This control scheme relies on an IIR-MLP predictor, which predicts the value of the control variables based on previously acquired samples, and on a classifier based on a static MLP NN. The classifier utilizes the predictions made by the IIR-MLP NN to recognise the kind of abnormal event that is about to occur on the network. High efficiency of the studied control scheme was reported.

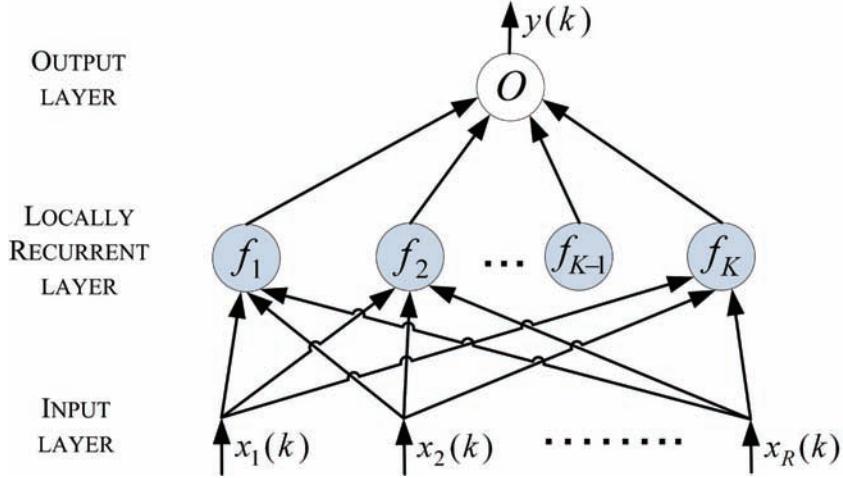
In Barbounis et al. (2006) the authors investigated the performance of IIR-MLP NN on the task of long-term wind speed and power forecasting based on meteorological information. Its performance was compared to the one of the local activation feedback multilayer network implemented with FIR synapses and the diagonal recurrent NN, as well as to the one of the FIR-MLP and static MLP NNs. The experimental results demonstrated that the IIR-MLP outperform all other architectures, paying back for the increased complexity.

In Cadini et al. (2007), IIR-MLP NN is employed for approximating the temporal evolution of a nonlinear dynamic system model of a simplified nuclear reactor. Specifically, the IIR-MLP NN is employed to estimate the neutron flux temporal evolution only knowing the reactivity forcing function. In a recent work, Cadini et al. (2008) employed the nonlinear modelling capabilities of IIR-MLP for tracking the nonlinear continuous-time dynamics of a nuclear reactor. The performance of the IIR-MLP is compared to the one of FIR-MLP and a buffered MLP NN. The authors report superior performance of the IIR-MLP NN.

Diagonal Recurrent Neural Networks

The diagonal recurrent neural network (DRNN) is a three layer network which possesses locally recurrent neurons in the hidden layer. It was formulated in (Ku et al., 1992; Ku & Lee, 1995), who studied the case where the locally recurrent neurons possess output feedbacks, and the output layer is composed of linear neurons. In the general case the locally recurrent neurons could be implemented as any of the

Figure 2. General architecture of a diagonal recurrent neural network



structures discussed in the previous section, and the output layer can be implemented as static or dynamic, depending on the application of interest.

The general architecture of DRNN is shown in Figure 2. Specifically, the inputs $x_i(k)$, $i=1,2,\dots,R$, are fed to all locally recurrent neurons f_j , $j=1,2,\dots,K$, forming the hidden layer of the network. In the general case, the number of inputs, R , is decoupled from the number of hidden neurons, K . The outputs of the locally recurrent layer are fed to the neuron(s) of the output layer, O_i , where the output of the DRNN, $y(k)$, is computed. In the specific implementation of the DRNN that Ku and Lee (1995) studied, the locally recurrent neurons possess static synaptic weights and the dynamic behaviour of the network is due solely to the output feedbacks of the hidden layer neurons. The neurons in the output layer, O_i , are implemented as linear neurons that sum the weighted outputs of the hidden layer neurons. For that specific implementation, the output neurons do not possess non-linear activation function and therefore the output, $y(k)$, of the DRNN is defined as:

$$y(k) = \sum_{j=1}^K w_j^O u_j(k) \\ u_j(k) = f_j \left(\sum_{i=1}^R w_{ij}^I x_i(k) + w_j^H u_j(k-1) \right), \quad (8)$$

where the output, $y(k)$, is a linear combination of the hidden layer outputs, $u_j(k)$, with $j=1,2,\dots,K$. Here, w_j^O , with $j=1,2,\dots,K$, are the static weights of the output layer neuron. Furthermore, the hidden layer output, $u_j(k)$, $j=1,2,\dots,K$, for the j th locally recurrent neuron is formed after applying the non-linear activation function, f_j (in the work of Ku and Lee (1995) this is the sigmoid activation function) on the sum of the weighted present inputs, $x_i(k)$, and the weighted past outputs, $u_j(k-1)$, of that neuron. Here, the weights w_{ij}^I , with $i=1,2,\dots,R$ and $j=1,2,\dots,K$, are the weights associated to the connection between the i th input and j th locally recurrent neuron, and w_j^H , with $j=1,2,\dots,K$, are the weights in the feedbacks of the locally recurrent neurons.

Recently, a second-order DRNN was studied in (Kazemy et al., 2007) as a natural extension of the first-order DRNN (Ku & Lee, 1995). In the second-order DRNN, each locally recurrent neuron possesses two feedbacks – the first one is from the previous state of its output $u_j(k-1)$, and the second one is from the state preceding it, i.e. from the state $u_j(k-2)$. The output of the second-order DRNN is computed as:

$$\begin{aligned} y(k) &= \sum_{j=1}^K w_j^O u_j(k) \\ u_j(k) &= f_j \left(\sum_{i=1}^R w_{ij}^I x_i(k) + w_j^{H1} u_j(k-1) + w_j^{H2} u_j(k-2) \right), \end{aligned} \quad (9)$$

where w_j^{H1} and w_j^{H2} are the weights in the first- and second-order feedbacks of each locally recurrent neuron. Due to the availability of this second feedback, i.e. extra memory directly providing information from previous state, the second-order DRNN was shown to approximate nonlinear functions better than other DRNNs. A higher order DRNN was studied in (Cho et al., 1997).

In the general case, all neurons in the hidden and output layers of the network can be implemented as locally recurrent neurons, and the static synaptic weights w_{ij}^I , w_j^H , and w_j^O , can be replaced by dynamic weights with transfer functions $G_{ij}^I(z^{-1})$, $H_{FBj}^H(z^{-1})$, $G_j^O(z^{-1})$, respectively. Therefore, for the general case equation (8) can be rewritten as:

$$\begin{aligned} y_l(k) &= O_l \left(\sum_{j=1}^K G_j^O(z^{-1}) u_j(k) + H_{FBl}^O(z^{-1}) y_l(k-1) \right) \\ u_j(k) &= f_j \left(\sum_{i=1}^R G_{ij}^I(z^{-1}) x_i(k) + H_{FBj}^H(z^{-1}) u_j(k-1) \right), \end{aligned} \quad (10)$$

where $H_{FBl}^O(z^{-1})$ are the transfer functions of the feedbacks of the output layer neurons. In the general case the transfer functions of the dynamic synaptic and feedback connections can be implemented with poles and zeroes as specified by equations (3) and (5).

In the work of Ku and Lee (1995), the DRNN were trained through a version of the back-propagation algorithm that the authors referred to as generalized dynamic back-propagation algorithm (DBP). Further details about the DBP algorithm, as well as the proof of its convergence and stability are available in (Ku & Lee, 1995).

Kazemy et al. (2007) studied a modification of the back-propagation training algorithm, which they introduced for training the proposed second-order DRNN. Convergence analysis of the proposed dynamic back-propagation algorithm was developed and the maximum learning rate that guarantees the convergence of the algorithm was derived.

Yu & Chang (2005) developed on-line adaptive training algorithm for the DRNN, which extends its use in nonlinear dynamic system modelling and model-based control.

Applications of the DRNNs

Initially, Ku et al. (1992) applied the DRNN for improved temperature control of nuclear reactor. In a later work, Ku and Lee (1995) developed DRNN-based identification and controller components for a control

system scenario. The performance of the DRNN and the proposed training method were investigated in a numerical evaluation, which involved five different scenarios. Among these were bounded-input bounded-output (BIBO) and non-BIBO non-linear plants and three related scenarios that evaluate the on-line adapting ability, the recovering ability and the interpolation ability of the DRNN-based control system. In the analysis Ku and Lee performed comparison of the proposed model with a feedforward neural network and with a fully connected recurrent neural network. The comparison had been performed in terms of number of required weights and ability for dynamic mapping characteristics. Analysing the results of the performed numerical evaluation, Ku and Lee came to the conclusion that DRNNs are applicable to on-line control applications. A higher order DRNN was used for identification of nonlinear dynamic systems (Cho et al., 1997).

Kasper et al., in (Kasper et al., 1995; Kasper et al., 1996), experimented with two-dimensional implementation of a DRNN. They compared their locally recurrent and sparsely connected structure with a fully connected recurrent neural network. Experimental evaluation on several linkage schemes for the DRNN, involving different size of the neighbourhood (this size was kept common for all neurons), had been performed. It was observed that the LRNN for neighbourhood size five and the fully recurrent neural networks achieve practically equivalent speech recognition accuracy.

Lai et al. (1999) employed a DRNN to control knee joint movement through stimulation of quadriceps muscle. Similarly to the work of Ku and Lee (1995), this control system contained a DRNN controller and a DRNN identifier. The neural controller calculated the required stimulus to activate the quadriceps muscle for a given trajectory. The DRNN identifier was used to learn the properties of the quadriceps-lower leg system.

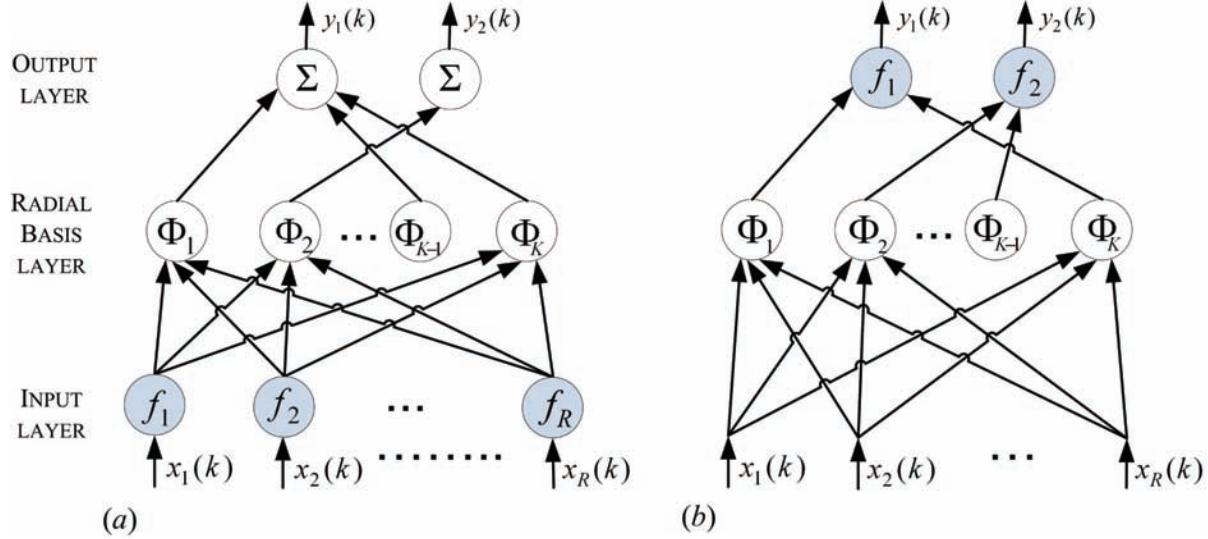
Jayawardhana et al. (2002) successfully used a DRNN for nonlinear mapping in active noise control (ANC) application. For the purpose of their application the authors modified the DRNN training. In the numerical evaluation, the DRNN expressed a better ability to deal with non-linear dynamic system than the feedforward structure. Evaluation of the computational demands and the performance of the DRNN for different training procedures were offered as well.

DRNN were used as controller for pneumatic robot servo system (Wang et al., 2003; Wang & Peng, 2003), for predictive control for active power filter (Shaosheng & Hui, 2004), for control of sonar array servo system (Liu & Du, 2005), etc.

Kazemy et al. (2007) applied a second-order DRNN in a system for real-time control of an image stabilization platform of a submarine periscope mirror. It was observed that the second-order DRNN provides more accurate identification and needs a shorter training time, when compared to the first-order DRNN.

Recently, Liu et al. (2008) employed DRNN in the design of an intelligent controller involved in trajectory tracking control of a human arm moving on the sagittal plane. The hybrid genetic algorithm and evolutionary program strategy was applied to optimize the DRNN architecture, and an adaptive dynamic back-propagation (ADBP) algorithm with momentum for the multi-input multi-output (MIMO) systems was used to obtain the network weights. The authors performed comparative evaluation of the DRNN performance against the one of the generalized iterative linear-quadratic regulator (ILQG) method, and the results demonstrated the advantage of DRNN.

Figure 3. Architecture of the locally recurrent radial basis function neural network: (a) with locally recurrent input layer; and (b) with locally recurrent output layer.



Locally Recurrent Radial Basis Function Neural Networks

The locally recurrent radial basis function neural networks (LRRBFNNs) are structures that combine the advantages of radial basis networks and the locally recurrent neurons discussed earlier. Here we focus our attention to two typical designs of LRRBFNN that employ the locally recurrent neurons either in the input layer or alternatively in the output layer of the network.

Figure 3 illustrates the two designs. Specifically, Figure 3 (a) shows a LRRBFNN that incorporates the locally recurrent neurons in the input layer. This structure was initially proposed by (Zemouri et al., 2003) as a way to incorporate memory about previous states into the RBF neural network. Like Zemouri et al., here we will consider only the Gaussian function, among the various implementations of the non-linear basis function that have been studied in the literature (Ripley, 1995). In the LRRBFNN proposed by Zemouri et al., the locally recurrent neurons in the first layer, as well as all other neurons in the network, possess synapses with static weights. Thus, the dynamic behaviour of the network is achieved solely by the local output feedbacks that enclose the locally recurrent neurons. In this case the output of the LRRBFNN, $y(k)$, can be expressed as:

$$y_l(k) = \sum_{j=1}^{K_l} w_j^o \exp\left(-\frac{\sum_{i=1}^R (u_i(k) - \mu_{ij})^2}{\sigma_j^2}\right) \quad (11)$$

$$u_i(k) = f_i(x_i(k) + w_i^I u_i(k-1)) ,$$

where w_j^o , with $j=1,2,\dots,K$, are the static weights of the output layer neuron, and the expression $\exp(\cdot)$ stands for the non-linear activation function of a RBF neuron. Here, σ_j^2 with $j=1,2,\dots,K$, defines the perceptive field of the j th RBF neuron, and μ_{ij} , with $i=1,2,\dots,R$ and $j=1,2,\dots,K$, is the i th centroid of that neuron. Furthermore, $u_i(k)$ is the output of the i th neuron in the locally recurrent layer, with $i=1,2,\dots,R$,

where the output, $u_i(k)$, of each locally recurrent neuron is formed as the sum of the input signal $x_i(k)$ and the weighted by the feedback weights w_i^I previous output of that neuron, $u_i(k-1)$. Finally, f_j stands for the non-linear activation function, which in the specific implementation of Zemouri et al. is the sigmoid function.

Figure 3 (b) shows another design of LRRBFNN, where the RBF neurons are in the hidden layer and the linear neurons with dynamic synaptic weights are positioned in the output layer of the network. This structure was introduced in (Ciocoiu, 1996; Ciocoiu, 1998), where the synaptic weights were implemented as dynamic FIR and IIR structures. The output of the neural network, $y(k)$, for this implementation of the LRRBFNN in time instant k is computed as:

$$\begin{aligned} y_l(k) &= f_l \left(\sum_{j=1}^{K_l} G_j(z^{-1}) u_j(k) \right) \\ u_j(k) &= \exp \left(-\frac{\sum_{i=1}^R (x_i(k) - \mu_{ij})^2}{\sigma_j^2} \right), \end{aligned} \quad (12)$$

where $G_j(z^{-1})$, with $j=1,2,\dots,K_l$, is the transfer function for the j th synapse of the l th neuron in the output layer, K_l is the total number of synapses that the l th neuron possess, and $u_j(k)$ is the output of the j th RBF neuron in the hidden layer. Similarly to the conventions in equation (11), σ_j^2 with $j=1,2,\dots,K_l$, defines the perceptive field of the j th RBF neuron, and μ_{ij} , with $i=1,2,\dots,R$ and $j=1,2,\dots,K_l$, is the i th centroid of that neuron.

In the general case both the input and output layers can possess locally recurrent neurons with dynamic synaptic weights. Then the output, $y_l(k)$, of the LRRBFNN for the l th neuron is expressed as:

$$\begin{aligned} y_l(k) &= f_l(H_{FBl}^O(z^{-1})y_l(k-1) + \sum_{j=1}^{K_l} G_j^O(z^{-1})\Phi_j(k)) \\ \Phi_j(k) &= \exp \left(-\frac{\sum_{i=1}^R (u_i^I(k) - \mu_{ij})^2}{\sigma_j^2} \right) \\ u_i^I(k) &= f_i(G_i^I(z^{-1})x_i(k) + H_{FBi}^Iu_i^I(k-1)), \end{aligned} \quad (13)$$

where $H_{FBl}^O(z^{-1})$ are the transfer functions of the feedbacks of the output layer neurons, $y_l(k-1)$ is the past output of l th neuron, $G_j^O(z^{-1})$ is the transfer function of the dynamic synaptic weight functions of the output neurons, $\Phi_j(k)$ is the output of the j th RBF neuron in the hidden layer, $u_i^I(k)$ is the output of the i th locally recurrent neuron in the input layer, $G_i^I(z^{-1})$ and $H_{FBi}^I(z^{-1})$ are the transfer functions of the synaptic and feedback weights for the i th neuron in the input layer, $x_i(k)$, with $i=1,2,\dots,R$, are the inputs, and $u_i^I(k-1)$ are the past outputs of the locally recurrent neurons in the input layer. The transfer functions of the dynamic synaptic and feedback connections can be implemented with poles and zeroes as specified by equations (3) and (5).

The LRRBFNNs (refer to equation (12)) have the following adjustable parameters: the number of RBF neurons in the hidden layer, K_l , the centres μ_{ij} and the receptive field σ_j^2 of each RBF, and the coefficients in the dynamic transfer function $G_j(z^{-1})$. The number of RBF neurons, as well as their means and receptive field can be trained by any of the algorithms outlined in (Back, 2001). Ciocoiu (1996, 1998)

proposed a LMS-type iterative learning algorithm for training the coefficients in the dynamic transfer function $G_j(z^{-1})$ of the LRRBFNN shown in Figure 3 (b). For that purpose, the instantaneous error in time instant k is estimated and then minimized by modifying the coefficients of the transfer function $G_j(z^{-1})$ in a direction opposite to the instantaneous gradients.

For the purpose of comprehensiveness we should also mention the work of Boquete et al. (1999b) which studies LRRBFNN with RBF neurons encompassed by local feedbacks implemented as FIR filters. A hybrid LRRBFNN, which combines RBF neurons with FIR feedbacks in the hidden layer and FIR synaptic weights in the summation layer, is studied in Boquete et al. (2005).

Applications of the LRRBFNNs

In (Ciocoiu, 1996) the proposed LRRBFNN was trained to simulate a low-pass Butterworth filter and the author reported competitive results, when compared to RBF neural network with locally connected fully recurrent layer of first order studied in (Mak, 1995). Later on, Ciocoiu (1998) studied the performance of the proposed LRRBFNN on chaotic time series analysis setup. The simulation results indicated that the proposed architecture is able to closely predict several future values of the process. It was observed that the performance improves for larger orders of the synaptic filters.

Zemouri et al. (2003) evaluated the proposed LRRBFNN on three time series predictions problems: MacKey-glass, logistic map, and Box & Jenkins gas furnace data. The authors concluded that the results obtained in the simulations validate the ability of the proposed neural network architecture to learn temporal sequences.

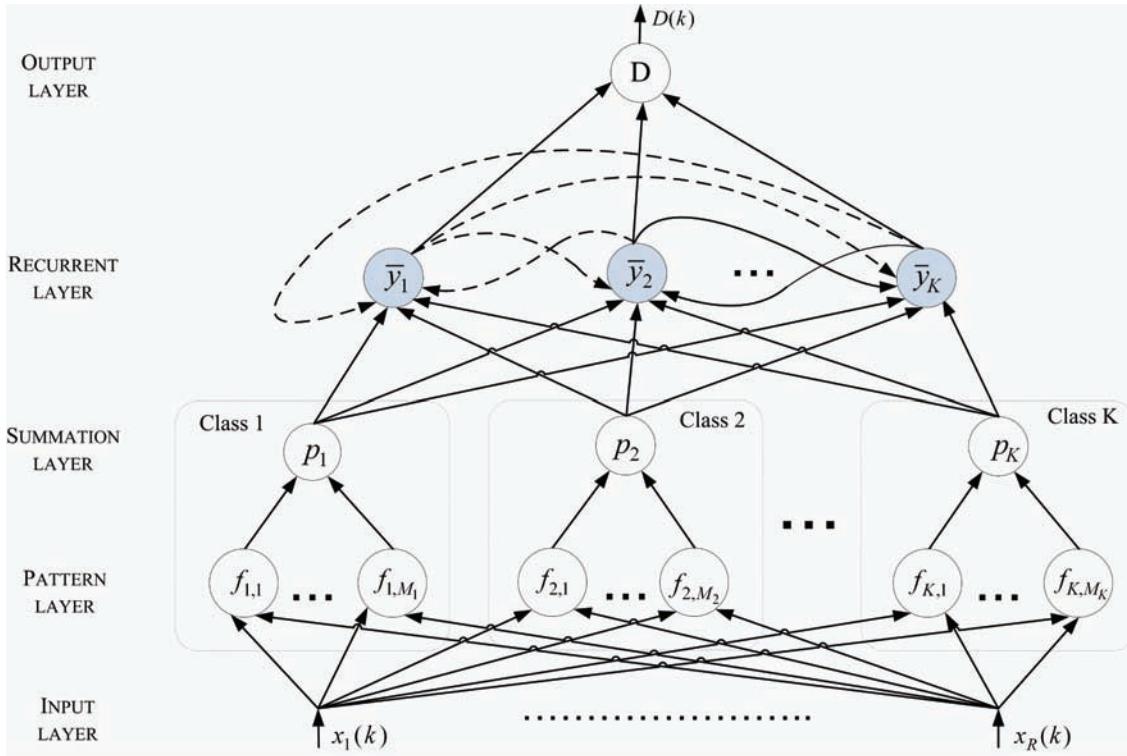
Boquete et al. (2005) employed a hybrid LRRBFNN, which possess local feedbacks encompassing each RBF neuron with a FIR filter as in Chen et al. (1993) but in addition the synaptic connections between the RBF neurons and the next layer summation units are implemented as FIR filters in the same manner as in Ciocoiu (1996). The hybrid LRRBFNN was employed in adaptive control loop, which aims at improving the basic PID-based speed control of a wheelchair. The authors demonstrated that the hybrid LRRBFNN has a number of advantages when compared to alternative solutions, reported in Boquete et al. (1999a), which utilizes a two-input two-output version of the LRRBFNN of Ciocoiu (1996), and in Boquete et al. (1999b) which relies on LRRBFNN whose RBF neurons are encompassed by local feedbacks implemented as FIR filters.

In a recent work, Gowrishankar & Satyanarayana (2008) employed a LRRBFNN with locally recurrent input layer for bit error rate prediction in a wireless local area network operating under the IEEE 802.11 standard. The authors modelled the wireless channel as time-varying non-linear system, whose behaviour is predicted by the LRRBFNN. Simulations were performed for different modulations and under multipath fading signals.

Partially Connected Locally Recurrent Probabilistic Neural Networks

The locally recurrent probabilistic neural networks (LRPNNs) introduced in (Ganchev et al., 2003; Ganchev et al., 2004), the generalized LRPNNs (Ganchev et al., 2007) and the partially connected LRPNN (PC-LRPNN) (Ganchev et al., 2008) belong to the huge class of locally connected recurrent neural networks. Specifically, the LRPNN was derived (Ganchev et al., 2003) from the original PNN (Specht, 1988) by incorporating an additional hidden layer, referred to as recurrent layer, between the summation layer and the output competitive layer of the PNN structure. This recurrent layer consists

Figure 4. Architecture of the partially connected locally recurrent probabilistic neural network



of locally recurrent neurons possessing local activation feedbacks from their past outputs as well as non-local feedbacks from all other neurons in that layer. Subsequently, Ganchev et al. (2004) studied alternative implementation, where the recurrent layer neurons possess local output feedbacks, which encompass the non-linear activation function of the recurrent layer neurons.

In the following discussion, we will consider the more general form of LRPNNs, referred to as partially connected LRPNNs, where the non-local links in the recurrent layer are learned from the input data or can be selected based on prior knowledge about the application of interest. In this way the recurrent layer neurons might possess, or might not possess, non-local recurrent connections to the other neurons in that layer. In the case when none of the possible non-local recurrent connections is implemented, the recurrent layer becomes populated only by locally recurrent neurons possessing self-feedbacks. These neurons can be viewed as the Frasconi-Gori-Soda activation feedback or output feedback types of locally recurrent structures (Frasconi et al., 1992).

Alternatively, when all possible non-local recurrent connections are implemented the recurrent layer becomes fully connected and the maximum memory capacity of the neural network is reached.

Figure 4 presents the simplified structure of a PC-LRPNN for classification in K classes. In contrast to the fully connected LRPNN, where each neuron in the recurrent layer communicates with all other neurons in that layer (i.e. global communication is enabled), in the PC-LRPNN the recurrent layer linkage is implemented only partially. This is illustrated in Figure 4, where the dashed line indicates that the linkage between neurons \bar{y}_1 and \bar{y}_2 and between \bar{y}_1 and \bar{y}_K might not be implemented. In general, the concept of partially connected recurrent layer can be regarded as defining local neighbourhoods for

each of the recurrent layer neurons. The local neural neighbors are not defined by the specific values of the neurons' indexes but are selected during training, on a competitive basis, and in data-dependent manner, with respect to certain predefined criterion. In practice, the size of neighbourhood and the recurrence depth (i.e. the depth of memory) in the recurrent layer are specified depending on prior knowledge about the specific problem at hand, or are identified heuristically after some experimentation with a representative dataset.

In the first two hidden layers the PC-LRPNNs, as their predecessor — the PNNs, implement the Parzen window estimator (Parzen, 1962) by using a mixture of Gaussian basis functions. If a PC-LRPNN for classification in K classes is considered, the class conditional probability density function $p_j(\mathbf{x}(k) | j)$, with $j=1,2,\dots,K$, for an input vector $\mathbf{x}(k)=\{\mathbf{x}_1(k), \mathbf{x}_2(k), \dots, \mathbf{x}_R(k)\}$ is defined as:

$$p_j(\mathbf{x}(k) | j) = f_j(\mathbf{x}_k) = \frac{1}{(2\pi)^{d/2} \sigma_j^d} \cdot \frac{1}{M_j} \sum_{i=1}^{M_j} \exp\left(-\frac{1}{2\sigma_j^2} (\mathbf{x}_k - \mathbf{x}_{ij})^T (\mathbf{x}_k - \mathbf{x}_{ij})\right), \quad j = 1, 2, \dots, K, \quad (14)$$

where for simplicity of further notations $p_j(\mathbf{x}(k) | j)$ is replaced by $f_j(\mathbf{x}_k)$. Here \mathbf{x}_{ij} is the i th training vector from class j , $\mathbf{x}(k)$ belonging to the set $\mathbf{X}=\{\mathbf{x}(k)\}$, with $k=1,2,\dots,P$, is the k th input vector, which is also designated as \mathbf{x}_k , d is the dimension of the input vectors, and M_j is the number of training patterns in class j . Each training vector \mathbf{x}_{ij} is assumed a centre of a kernel function, and consequently the number of pattern units in the first hidden layer of the neural network is given by the sum of the pattern units for all the classes. The standard deviation σ_j acts as a smoothing factor, which softens the surface defined by the multiple Gaussian functions. Instead of the simple covariance matrix, $\{\sigma_j^2 I\}$, where I represents the identity matrix, the full covariance matrix can be computed using the Expectation Maximization algorithm, as proposed in (Yang & Chen, 1998; Mak & Kung, 2000) and elsewhere. Here for simplicity of exposition, we consider the simple case, where the value of the standard deviation is identical for all pattern units belonging to a specific class. Moreover, σ_j can be the same for all pattern units irrespective of their class belonging, as it was originally proposed in (Specht, 1990).

Next, the class conditional probability density functions $f_j(\mathbf{x}_k)$ for each class j , estimated through equation (14), act as inputs for the recurrent layer. The recurrent layer is composed of locally recurrent neurons, which in addition to the inputs coming from the summation layer and their own past outputs, also might possess non-local feedbacks from current and past outputs of the neurons for the other classes that belong to their neighbourhood.

The summation units' output, $y_j(\mathbf{x}_k)$, of the neurons in the locally connected recurrent layer is computed as:

$$y_j(\mathbf{x}_k) = b_{jj} f_j(\mathbf{x}_k) - \sum_{\substack{i=1 \\ i \neq j}}^K b_{ij} f_i(\mathbf{x}_k) + \sum_{t=1}^N (a_{ijt} \bar{y}_j(\mathbf{x}_{k-t}) - \sum_{\substack{i=1 \\ i \neq j}}^K a_{ijt} \bar{y}_i(\mathbf{x}_{k-t})), \quad j = 1, 2, \dots, K, \quad (15)$$

where $f_j(\mathbf{x}_k)$ is the probability density function of each class j , \mathbf{x}_k is the k th input vector, K is the number of classes, N is the recurrence depth, $\bar{y}_j(\mathbf{x}_{k-t})$ is the normalized past output for class j that has been delayed on t time instants, and a_{ijt} and b_{ij} are weight coefficients. The two indexes of the weights b_{ij} , with $j=1,2,\dots,K$ and $i=1,2,\dots,K$, stand for the current recurrent neuron and for the class to which the

corresponding input belongs. The first two indexes of the weights a_{ijt} have the same meaning as for the weights b_{ij} , and the third index $t=1,2,\dots,N$ indicates the time delay of the specific output before it appear as an input.

In general, the locally connected recurrent layer can be considered as an IIR filter that smoothes the probabilities generated for each class by exploiting one or more past values of its output, and potentially by incorporating present and past information for some or all competitive classes.

The output, $y_j(\mathbf{x}_k)$, of each summation unit of a neuron in the recurrent layer is subject to the normalization:

$$\bar{y}_j(\mathbf{x}_k) = \frac{\text{sgm}(y_j(\mathbf{x}_k))}{\sum_{i=1}^K \text{sgm}(y_i(\mathbf{x}_k))}, \quad j = 1, 2, \dots, K, \quad (16)$$

which retains the probabilistic interpretation of the output of the recurrent layer, i.e. $\sum_{j=1}^K \bar{y}_j(\mathbf{x}_k) = 1$, for the k th input vector \mathbf{x}_k . Here, the designation $\text{sgm}(\cdot)$ refers to the sigmoid activation function.

Subsequently, in the output layer, often referred to as competitive layer, the Bayes optimal decision rule (17) is applied to distinguish class j , to which the input vector \mathbf{x}_k is categorized:

$$D(\mathbf{x}_k) = \underset{j}{\operatorname{argmax}} \{h_j c_j \bar{y}_j(\mathbf{x}_k)\}, \quad j = 1, 2, \dots, K, \quad (17)$$

where h_j is a priori probability of occurrence of a pattern from class j , and c_j is the cost function associated with the misclassification of a vector belonging to class j .

In a recent work, Ganchev et al. (2007) introduced the generalized LRPNN (GLRPNN) which possesses more memory units in the recurrent layer than any other LRPNNs. Specifically, in contrast to LRPNNs, GLRPNNs allow delayed outputs of the summation layer, i.e. probabilities estimated at past time instants, to be fed directly at the input of the recurrent layer neurons. This increased memory capacity allowed more complex mapping between inputs and outputs, which resulted in an improved classification capability. The price to pay is an increased number of weights to train.

Similarly to PC-LRPNNs, GLRPNNs can be implemented as a partially connected structure, where portion of the non-local recurrent connections in the recurrent layer are not implemented. This development would result in a locally connected recurrent layer which is populated with partially or fully linked groups of recurrent neurons and a quantity of locally recurrent neurons which are not connected with the other neurons in that layer. When none of the non-local recurrent connections are implemented, the locally connected recurrent layer is transformed to locally recurrent layer, populated entirely by locally recurrent neurons of the type referred to as generalized Frasconi-Gori-Soda architecture (Tsoi & Back, 1994), shown in Figure 1(c). These recurrent neurons could be implemented either with local activation feedbacks as in (Ganchev et al., 2003), or with local output feedbacks as in (Ganchev et al., 2004).

The LRPNNs with local activation feedbacks in the recurrent neurons (Ganchev et al., 2003) and these with local output feedbacks (Ganchev et al., 2004) were trained via common three-step procedure. The first two steps build the pattern layer and estimate the value of the smoothing factor σ_j , while the third training step adjusts the recurrent layer weights through iterative differential evolution-based scheme.

The PC-LRPNN is trained via four-step procedure (Ganchev et al., 2008). Similarly to the training

of LRPNNs, the first two steps build the pattern layer and estimate the value of the smoothing factor σ_j . The third training step selects the recurrent layer linkage to be implemented, and the fourth one adjusts the weights of the recurrent neurons via PSO-based training scheme.

In conclusion, we would like to emphasize that in the general case, the family of LRPNNs implements non-local feedback connections among the recurrent layer neurons. These connections link up the recurrent neuron for a given class with the recurrent neurons of the competitive classes. These links offer information about the status of the competitor classes that can be exploited for improving the contextual interpretation of the inputs for the own class. Depending on the situation these non-local feedbacks from the competitors act either as inhibitors or amplifiers of the neuron's own input. Somehow, these relationships among the artificial neurons in a neural neighbourhood bear similarity to what we imagine to happen in neighbouring columns of biological neurons in the cortex. Judging from this perspective, the PC-LRPNNs, due to their recurrent layer that might incorporate both locally recurrent neurons and non-locally recurrent neural neighbourhoods, can be thought to as a more advanced structure that offers a better biological plausibility than the other LRNNs discussed in the previous sections.

Applications of the LRPNNs

Ganchev et al. (2003, 2004) employed LRPNNs on the text-independent speaker verification problem. Specifically, for each target user a speaker-devoted LRPNN was trained to recognize him/her against the rest of the world, which was represented by a purposely built reference model. Significant improvement of performance was reported for LRPNN with recurrence depth of one, two and three, i.e. with feedbacks from time instants $k-1$, $k-2$ and $k-3$, when compared to the static PNN.

In Ganchev et al. (2007) the GLRPNN, i.e. a LRPNN with fully connected recurrent layer and fully buffered inputs, was compared against FIR, IIR, DRNN and LRPNN structures. In numerical experiments on the text-independent speaker verification problem, the GLRPNN and LRPNN outperformed the other structures of interest.

Recently, Ganchev et al. (2008) employed the PC-LRPNN on the text-independent speaker identification and on the speaker-independent emotion recognition problems. On both problems, a single PC-LRPNN-based classifier was built to distinguish among the categories of interest. It was observed that PC-LRPNNs significantly outperform the PNN-, LRPNN-, and Gaussian mixture models (GMM)-based classifiers.

Quantitative Account of a Single Layer of Neurons

Let's consider a single layer of neurons, which doesn't possess feedbacks from other layers. We can estimate the complexity, i.e. the number of weights/links that different types of neurons populating that layer would result in. Each type of neurons considered in the following could be derived from the general structure shown in Figure 1 (c), with a certain number of modifications.

In the general case, the aforementioned layer can possess K neurons, with M inputs each. Each input is weighted by dynamic synaptic function, G_i , as formulated in equation (3). The transfer function of each synaptic weight can have $p=n_p+m_z+2$ coefficients. Similarly the L local recurrent feedbacks are weighted by transfer function, H_{FB} , defined in equation (5) that can possess $r=g_z+l_p+2$ coefficients. Finally, there could be N non-local recurrent feedbacks, each weighted by a dynamic weight that has

transfer function with q coefficients. In this very general case, the total number of weights that the layer of neurons could possess can be expressed as:

$$S = K(pM + rL + qN + 1) \quad (18)$$

where the last term “1” accounts for the bias weight for each of the K neurons.

Depending on the type of neurons employed in the design of this layer, we can estimate the total number of weights to train as follows:

- (i) For a layer implemented with FIR neurons: Since FIR structures do not possess feedbacks from past states, i.e. $L=0$ and $N=0$, the total number of weights is

$$S_{FIR} = K(pM + 1), \quad (19)$$

with $p > 1$. In the case of static synaptic weights $p=1$, and therefore $S_{FIR} = K(M+1)$.

- (ii) For a layer implemented with IIR neurons: The IIR structures possess local recurrent feedbacks but do not possess non-local recurrent feedbacks, i.e. $N=0$. Then, the total number of weights is

$$S_{IIR} = K(pM + rL + 1), \quad (20)$$

with $p > 1$, $r > 1$. In the case of static synaptic weights $p=1$ and $r=1$, and therefore $S_{IIR} = K(M+L+1)$.

- (iii) For a locally connected recurrent layer: All neurons possess local feedbacks and some number, N , of non-local recurrent feedbacks, with $L(K-1) > N > 0$. The last reflects the case where some of the possible non-local connections are not implemented. Then, the total number of weights is

$$S_{LC} = K(pM + rL + qN + 1), \quad (21)$$

with $p > 1$, $r > 1$, and $q > 1$. In the case of static synaptic weights we have $p=1$, $r=1$, and $q=1$, and therefore $S_{LC} = K(M+L+N+1)$.

- (iv) For a fully connected recurrent layer: All neurons possess local recurrent feedbacks and all possible non-local recurrent feedbacks, with $N=L(K-1)$. Then the total number of weights is

$$S_{FC} = K(pM + L(r+q(K-1))+1), \quad (22)$$

with $p > 1$, $r > 1$, and $q > 1$. In the case of static synaptic weights we have $p=1$, $r=1$, and $q=1$, and therefore $S_{FC} = K(M+LK+1)$.

Obviously, under the condition that p , r , and q are kept constant across the different implementations, we observe that $S_{FC} > S_{LC} > S_{IIR} > S_{FIR}$ and $S_{FC} > S_{LC} > S_{IIR} > S_{FIR}$. Thus, when the type of neurons for specific application has to be selected, one must consider the total number of adjustable weights to estimate and the amount of available training data.

Just for being more illustrative, let's consider an intuitive example, which offers a brief account of the number of weights to train for a layer of neurons that possesses only static synaptic weights. In this simplified example, we will assume that the M inputs to that layer have unit weights, and therefore their adjustment is not necessary. For further simplification we can omit the estimation of the bias weights.

Thus, we consider equations (19)-(22) for the case of static synaptic weights and with $M=0$ and no bias weights. Thus, according to equation (22), and as it is well known from the literature, a fully recurrent layer of first order, i.e. $L=1$ with K neurons would possess approximately K^2 weights to train. For a higher order network it could be that each neuron receives multiple delayed feedbacks from all past states of the rest of the neurons. Then the total number of weights would be approximately LK^2 , where L is the recurrence depth. On the other hand, if the neurons possess non-local connections only from the most recent output of the other neurons, then the number of weights is only $(L+K)K$.

For a locally connected recurrent layer (refer to equation (21)) obviously these estimations will be reduced depending on what part of the linkage is implemented (let's remember that $L(K-1)>N>0$).

Next, according to equation (20), for a locally recurrent layer of the diagonal type the approximate number of weights to train would be LK .

Finally, in the case of feedforward architecture, equation (19), we need to train only the M synaptic weights of the inputs plus the bias weights that we already ignored in the beginning of this example.

On the other hand, for an equal number of weights, the aforementioned layer exhibits dissimilar dynamic properties, which depend on the implementation of its linkage. Thus, for the selection of proper neuron type it is required not only an appropriate estimation of the necessary memory capacity but knowledge about the behavior of the network in each particular implementation. A brief account for the number of links and the number of delay elements (memory units) for a three-layer network that possesses only static synaptic weights is available in (Lin et al., 1998). They evaluated the behaviour of several architectures, possessing similar number of connections and investigated their performance on the latching problem and a grammatical inference problem. Their study illustrated one particular case on how the memory capacity depends on linkage and the number of feedbacks.

FUTURE TRENDS

Although the research on locally recurrent neural networks holds a proven record of successful applications, we deem that the potential of advance it proffers has not been developed fully, yet. Moreover, in the last few years, we observed a resumption of interest to the field of LRNNs, which resulted into a number of new neural network paradigms that emerged recently. These new architectures are in anticipation of comprehensive studies and fine-tuning for the needs of contemporary and traditional real-world applications. The process of application development, integration and deployment, as well as the operation in real-world conditions, would definitely stimulate further improvement and elaboration of the existing models, or alternatively might inspire innovative designs.

Speaking generally, in the past decade various studies demonstrated that the FIR- and IIR-based locally recurrent neural networks are able to model the Chua's circuits and to simulate the behaviour of autonomous non-linear systems. Furthermore, DRNNs and LRRBFNNs have been studied comprehensively and a variety of modifications of the network architecture, or its training, for particular problems, were proposed. Numerous successful applications of the LRNNs have been already developed, and presently operate on immense diversity of tasks (industrial control, signal processing, health and biological applications, weather modeling and forecasting, etc). Nevertheless, there is a well placed confidence in the capacity of LRNNs, and their derivative architectures, to extend the range of problems that they presently resolve, especially when they are implemented in their dynamic versions.

One interesting and promising direction for future research would be the exploration of composite and hybrid architectures, which incorporate either DRNNs or LRRBFNNs together with fuzzy networks and other successful approaches. Structures that combine locally connected and locally recurrent structures have the potential to offer new functionality that the “pure” locally recurrent neural networks cannot cover.

Regarding the family of LRPNNs, we can point out the compelling need of further studies that will investigate comprehensively existing and new approaches for optimization of the recurrent layer linkage, and its automatic adaptation for the requirements of particular application scenarios. Perhaps, new strategies for automatic selection of neighborhood size and the most appropriate neighbors of each recurrent neuron that arise directly from the training data will appear. Furthermore, any research that promotes the development of dynamically changing neighborhoods, which depend on the on-line inputs during the operation of the neural network, would be of great interest.

Along with the theoretical interest in new architectures and advanced training algorithms, we acknowledge the great importance of practical implementation and proliferation of the LRNNs and the neural networks in general. In this direction, the recent material implementation of the *memristor* (Strukov et al., 2008), a circuit element with nonlinear behavior that provides a functional relation between electrical charge and flux, brings forward new opportunities for development of neural network-based applications. As Chua and Kang (1976) demonstrated the computational model of artificial neuron can be modeled through the memristive equations and this result offers a wide avenue for cost-efficient implementation of a large number of neural network architectures in semiconductor-based integrated circuits.

Finally, despite the progress made during the past decades, we deem that the locally recurrent neural networks still await their golden time, when they will benefit from significantly better biological plausibility. The structure and the operation of the human brain is still a source of precious knowledge and inspiration, and we are looking forward to see how the development of Neurosciences will contribute for the further progress in the field of neural networks.

CONCLUSION

In order to summarize the properties of the locally recurrent neural networks considered here, we list their major advantages with respect to their non-recurrent counterparts, as well as to the buffered MLP’s, partially and fully connected recurrent neural networks. According to Campolucci et al., (1999), the advantages of LRNNs can be formulated as:

- (i) well-known neuron interconnection topology, i.e., the efficient and hierachic multilayer; this opposes to the sometimes non-obvious structure of the fully connected recurrent neural networks.
- (ii) small number of neurons required for a given problem, due to the use of powerful dynamic neuron models; this opposes to the static feedforward architectures.
- (iii) generalization of the popular FIR-MLP (or time-delay neural networks (TDNN)) to the infinite memory case;
- (iv) prewired forgetting behavior, needed in applications such as digital signal processing (DSP), system identification and control; i.e. the availability of memory in comparison to the memory-less feedforward architectures.
- (v) simpler training than fully recurrent networks;

- (vi) many training algorithms could be derived from filter theory and recursive identification.

Depending on the problem at hand and on the accepted point of view, some of these could be also thought to as disadvantages. For instance, one can argue that the locally connected and fully connected neural networks possess greater memory capacity than LRNNs. Therefore in cases where an abundance of training data is available, the fully recurrent and locally connected structures would offer either more compact solutions or advantageous performance, when compared to the LRNNs.

Furthermore, the availability of non-local connections within the locally connected and partially connected recurrent neural networks enable for exploiting sources of information, for which the LRNNs are blind. As one illustrative example we can point out the availability of non-local connections in the neural neighbourhood (or with all neurons from the recurrent layer) that is implemented in the family of LRPNNs. Such connections among the recurrent neurons provide information about the status of the competitive classes, and this information is exploited for a more accurate interpretation of the input to the own class, and thus for making more appropriate decisions.

Summarizing the discussion in this section we would like to emphasize that the LRNNs, the LRPNNs, the partially and fully connected recurrent neural networks have their specific properties, which one may interpret as advantages or disadvantages, depending on the requirements of every particular application. In that way, depending on the desired use, each one of the aforementioned architectures has its strengths and weaknesses, which define their area of usefulness.

EPILOGUE

Nowadays we all benefit from the vastly expanding Internet, which established new standards for information and service delivery. The dynamic nature of Internet also brings some inconveniences, such as missing or outdated web-pages, expired/broken links, etc. However, these lines are about the few initiatives that endured during the years and became indispensable sources of knowledge about the theory and practice of neural networks, and that attract the interest of students, hobbyists, developers and researchers. Undoubtedly, the most valuable and most useful among these initiatives is the newsgroup *comp.ai.neural-nets*. This forum, sustained by the neural network community, offers multiple benefits, such as knowledge exchange, sharing ideas and opinions, information about new publications, software, calls for papers, etc. This is the place where the interested reader of this book can receive answers of his questions, when they are related to theory of neural networks and their applications. This is the place to initiate a discussion on topic correlated to neural networks or to ask “How to ...?” questions.

Furthermore, a systematic catalogue of useful information is available in the knowledge repository, entitled “*Frequently Asked Questions*” (search on Internet for: comp.ai.neural-nets FAQ). Its impact comes from the comprehensive and systematic information, organized in chapters such as: Introduction, Learning, ..., Books, data, etc, Free/Commercial software, Applications, etc. This is the place, where everyone who has a question should check first before sending a post to the aforementioned newsgroup.

Wikipedia, a fast growing and well maintained knowledge repository that relies on voluntary contributions, is where the interested reader of this book can search for systemized information about the architecture, training or application of neural networks. This is the place where information is collected on a daily basis so if the needed answers are not available today, they may appear there tomorrow.

REFERENCES

- Back, A. (2001). Radial basis functions. In Y. H. Hu & J.-N. Hwang (Eds.), *Handbook of neural network signal processing*. London: CRC Press.
- Back, A. D., & Tsoi, A. C. (1991). FIR and IIR synapses, a new neural network architecture for time series modeling. *Neural Computation*, 3, 375–385. doi:10.1162/neco.1991.3.3.375
- Back, A. D., & Tsoi, A. C. (1992). Nonlinear system identification using multilayer perceptrons with locally recurrent synaptic structure. In *Proceedings of the 1992 IEEE-SP Workshop on Neural Networks for Signal Processing* (Vol. 2, pp. 444-453).
- Barbounis, T. G., Theocharis, J. B., Alexiadis, M. C., & Dokopoulos, P. S. (2006). Long-term wind speed and power forecasting using local recurrent neural network models. *IEEE Transactions on Energy Conversion*, 21(1), 273–284. doi:10.1109/TEC.2005.847954
- Barbounis, T. G., & Theocharis, J. B. (2007a). A locally recurrent fuzzy neural network with application to the wind speed prediction using spatial correlation. *Neurocomputing*, 70(7-9), 1525–1542. doi:10.1016/j.neucom.2006.01.032
- Barbounis, T. G., & Theocharis, J. B. (2007b). Locally recurrent neural networks for wind speed prediction using spatial correlation. *Information Sciences*, 177(24), 5775–5797. doi:10.1016/j.ins.2007.05.024
- Bengio, Y., de Mori, R., & Gori, M. (1992). Learning the dynamic of speech with back-propagation for sequences. *Pattern Recognition Letters*, 13, 375–385. doi:10.1016/0167-8655(92)90035-X
- Boquete, L., Barea, R., García, R., Mazo, M., & Espinosa, F. (1999a). Identification and control of a wheelchair using recurrent neural networks. *Engineering Applications of Artificial Intelligence*, 12(4), 443–452. doi:10.1016/S0952-1976(99)00021-4
- Boquete, L., García, R., Barea, R., & Mazo, M. (1999b). Neural control of the movements of a wheelchair. *Journal of Intelligent & Robotic Systems*, 25(3), 213–226. doi:10.1023/A:1008068322312
- Boquete, L., Barea, R., García, R., Mazo, M., & Sotelo, M. A. (2005). Control of a robotic wheelchair using recurrent networks. *Autonomous Robots*, 18, 5–20. doi:10.1023/B:AURO.0000047285.40228.eb
- Cadini, F., Zio, E., & Pedroni, N. (2007). Simulating the dynamics of the neutron flux in a nuclear reactor by locally recurrent neural networks. *Annals of Nuclear Energy*, 34, 483–495. doi:10.1016/j.anucene.2007.02.013
- Cadini, F., Zio, E., & Pedroni, N. (2008). Validation of infinite impulse response multilayer perceptron for modelling nuclear dynamics. *Science and Technology of Nuclear Installations*, 2008, 1–10. doi:10.1155/2008/681890
- Campolucci, P., Uncini, A., Piazza, F., & Rao, B. D. (1999). On-line learning algorithms for locally recurrent neural networks. *IEEE Transactions on Neural Networks*, 10(2), 253–271. doi:10.1109/72.750549
- Campolucci, P., Uncini, A., & Piazza, F. (1996, May). Fast adaptive IIR-MLP neural networks for signal processing application. In *Proceedings of the IEEE International Conference on Acoustic Speech and Signal Processing, ICASSP'96*, Atlanta, USA (pp. 3530-3533).

- Cannas, B., Cincotti, S., Fanni, A., Marchesi, M., Pilo, F., & Usai, M. (1998a). Performance analysis of locally recurrent neural networks. *COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, 17(6), 708–716. doi:10.1108/03321649810221251
- Cannas, B., Celli, G., Marchesi, M., & Pilo, F. (1998b). Neural networks for power system condition monitoring and protection. *Neurocomputing*, 23, 111–123. doi:10.1016/S0925-2312(98)00065-4
- Ciocoiu, I. B. (1996). RBF networks with FIR/IIR synapses. *Neural Processing Letters*, 3(1), 17–22. doi:10.1007/BF00417785
- Ciocoiu, I. B. (1998). Time series analysis using RBF networks with FIR/IIR synapses. *Neurocomputing*, 20, 57–66. doi:10.1016/S0925-2312(98)00024-1
- Chan, L. W., & Young, E. F.-Y. (1995, June). *Locally connected recurrent networks* (Tech. Rep. CS-TR-95-10). Hong Kong: The Chinese University of Hong Kong New Territories, Computer Science Department.
- Chen, C.-L., Chen, W.-C., & Chang, F.-Y. (1993). Hybrid learning algorithm for Gaussian potential function networks. *IEEE Proceedings. Part D. Control Theory and Applications*, 140(6), 442–448.
- Cho, J. S., Kim, Y. W., & Park, D. J. (1997). Identification of nonlinear dynamic systems using higher order diagonal recurrent neural network. *IEEE Electronics Letters*, 33(25), 2133–2135. doi:10.1049/el:19971398
- Chua, L. O., & Kang, S. M. (1976). Memristive devices and systems. *Proceedings of the IEEE*, 64, 209–223. doi:10.1109/PROC.1976.10092
- Chua, L. O. (1993). Global unfolding of Chua's circuit. *IEICE Transactions on Fundamentals E (Norwalk, Conn.)*, 76-A, 704–734.
- Chua, L. O., & Lin, G. N. (1990). Canonical realisation of Chua's circuit family. *IEEE Transactions on Circuits and Systems*, 37(7), 885–902. doi:10.1109/31.55064
- Chua, L. O., Komuro, M., & Matsumoto, T. (1986). The double scroll family. *IEEE Transactions on Circuits and Systems*, 33(11), 1072–1118. doi:10.1109/TCS.1986.1085869
- Elizondo, D., & Fiesler, E. (1997). A survey of partially connected neural networks. *International Journal of Neural Systems*, 8(5-6), 535–558. doi:10.1142/S0129065797000513
- Frasconi, P., Gori, M., & Soda, G. (1992). Local feedback multilayered networks. *Neural Computation*, 4, 120–130. doi:10.1162/neco.1992.4.1.120
- Ganchev, T., Tasoulis, D. K., Vrahatis, M. N., & Fakotakis, N. (2003, September). Locally recurrent probabilistic neural network for text-independent speaker verification. In *Proceedings of the 8th European Conference on Speech Communication and Technology, EUROSPEECH 2003* (Vol. 3, pp. 1673–1676).
- Ganchev, T., Tasoulis, D. K., Vrahatis, M. N., & Fakotakis, N. (2004). Locally recurrent probabilistic neural networks with application to speaker verification. *GESTS International Transaction on Speech Science and Engineering*, 1(2), 1–13.

- Ganchev, T., Tasoulis, D. K., Vrahatis, M. N., & Fakotakis, N. (2007). Generalized locally recurrent probabilistic neural networks with application to text-independent speaker verification. *Neurocomputing*, 70(7-9), 1424–1438. doi:10.1016/j.neucom.2006.05.012
- Ganchev, T., Parsopoulos, K. E., Vrahatis, M. N., & Fakotakis, N. (2008). Partially connected locally recurrent probabilistic neural networks. In X. Hu & P. Balasubramaniam (Eds.), *Recurrent neural networks* (pp. 377-400). Vienna, Austria: ARS Publishing.
- Gori, M. (1989, November). An extension of BPS. In *Proceedings of the 2nd International Workshop on Neural Networks and their Applications*, Nimes, France (pp. 83-93).
- Gori, M., Bengio, Y., & de Mori, R. (1989). BPS: A learning algorithm for capturing the dynamic nature of speech. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN'89* (Vol. 2, pp. 417-423).
- Gori, M., & Soda, G. (1990). Temporal pattern recognition using EBPS. In *Proceedings of the EURASIP Workshop 1990 on Neural Networks* (LNCS 412, pp. 69-80). London, UK: Springer-Verlag.
- Gowrishankar, , & Satyanarayana, P. S. (2008). Recurrent neural network based bit error rate prediction for 802.11 wireless local area network . *International Journal of Computer Sciences and Engineering Systems*, 2(3), 177-184.
- Grassi, G., & Cafagna, D. (2003). Locally-connected neural networks: Stability analysis and synthesis technique. In N. Mastorakis (Ed.), *Recent advances in intelligent systems and signal processing* (pp. 321-326).
- Jayawardhana, B., Xie, L., & Yuan, S. (2002, August). Active control of sound based on diagonal recurrent neural network. In *Proceedings of the Conference of the Society of Instrument and Control Engineers (SICE 2002)*, Osaka, Japan (pp. 2666-2671).
- Kasper, K., Reininger, H., Wolf, D., & Wust, H. (1995). A speech recognizer based on locally recurrent neural networks. In *Proceedings of the International Conference on Artificial Neural Networks*, 2, 15–20.
- Kasper, K., Reininger, H., & Wust, H. (1996). Strategies for reducing the complexity of a RNN-based speech recognizer. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing*, 6, 3354–3357. doi:10.1109/ICASSP.1996.550596
- Kazemy, A., Hosseini, S. A., & Farrokhi, M. (2007, June). Second order diagonal recurrent neural network. In *Proceedings of the IEEE International Symposium on Industrial Electronics, ISIE, 2007*, 251–256.
- Ku, C. C., & Lee, K. Y. (1995). Diagonal recurrent neural networks for dynamic system control. *IEEE Transactions on Neural Networks*, 6(1), 144–156. doi:10.1109/72.363441
- Ku, C. C., Lee, K. Y., & Edwards, R. M. (1992). Improved nuclear reactor temperature control using diagonal recurrent neural networks. *IEEE Transactions on Nuclear Science*, 39(6), 2298–2308. doi:10.1109/23.211440

- Lai, J. S., Luh, J. J., Lee, J. F., Chen, S. C., & Kuo, T. S. (1999, August). A DRNN based FES controller for joint position control – a computer study. In *Proceedings of the 4th Annual Conference of the International Functional Electrical Stimulation Society*, Sendai, Japan.
- Lapedes, A., & Farber, R. (1987). *Nonlinear signal processing using neural networks prediction and system modeling* (Tech. Rep. LA-UR-262). Los Alamos, USA: Los Alamos National Laboratory.
- Leighton, R. R., & Conrath, B. C. (1991). The autoregressive backpropagation algorithm. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN, ii*, 369–377. doi:10.1109/IJCNN.1991.155362
- Lin, T., Horne, B. G., & Giles, C. L. (1998). How embedded memory in recurrent neural network architectures helps learning long-term temporal dependencies. *Neural Networks*, 11, 861–868. doi:10.1016/S0893-6080(98)00018-5
- Liu, S., & Du, Y. (2005). Sonar array servo system based on diagonal recurrent neural network. In *Proceedings of the IEEE International Conference on Mechatronics & Automatics*, Niagara Falls, Ontario, Canada (pp. 1912-1917).
- Liu, S., Wang, Y., & Zhu, Q. M. (2008). Development of a new EDRNN procedure in control of human arm trajectories. *Neurocomputing*, 72, 490–499. doi:10.1016/j.neucom.2007.12.012
- Mandic, D. P., & Chambers, J. A. (2001). *Recurrent neural networks for prediction*. Chichester, UK: John Wiley & Sons Ltd.
- Mak, M. W. (1995). A learning algorithm for recurrent radial basis functions networks. *Neural Processing Letters*, 2(1), 27–31. doi:10.1007/BF02312380
- Mak, M. W., & Kung, S. Y. (2000). Estimation of elliptical basis function parameters by the EM algorithm with application to speaker verification. *IEEE Transactions on Neural Networks*, 11(4), 961–969. doi:10.1109/72.857775
- McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5, 115–133. doi:10.1007/BF02478259
- McDonnell, J. R., & Waagen, D. (1994). Evolving recurrent perceptrons for time-series modeling. *IEEE Transactions on Neural Networks*, 5(1), 24–38. doi:10.1109/72.265958
- Mozer, M. C. (1989). A focused backpropagation algorithm for temporal pattern recognition. *Complex Systems*, 3, 349–381.
- Mozer, M. C. (1993). Neural net architectures for temporal sequence processing. In A. Weigend & N. Gershenfeld (Eds.), *Predicting the future and understanding the past*. Redwood City, CA, USA: Addison Wesley.
- Natschläger, T., Maass, W., Sontag, E. D., & Zador, A. (2000, November). Processing of time series by neural circuits with biologically realistic synaptic dynamics. In *Proceedings of the Neural Information Processing Systems, NIPS 2000*, Denver, Colorado, USA (pp. 145-151).

- Natschläger, T., Maass, W., & Zador, A. (2001). Efficient temporal processing with biologically realistic dynamic synapses. *Network (Bristol, England)*, 12(1), 75–87. doi:10.1080/713663153
- Parzen, E. (1962). On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33, 1065–1076. doi:10.1214/aoms/1177704472
- Principe, J. C., de Vries, B., & de Oliveira, P. G. (1993). The gamma filter – a new class of adaptive IIR filters with restricted feedback. *IEEE Transactions on Signal Processing*, 41(2), 649–656. doi:10.1109/78.193206
- Psarrou, A., Shaosheng, G., & Buxton, H. (1995, November). Modelling spatio-temporal trajectories and face signatures on partially recurrent neural networks. In *Proceedings of the IEEE International Conference on Neural Networks*, Perth, Australia (Vol. 5, pp. 2226-2231).
- Ripley, B. D. (1995). *Pattern recognition and neural networks*. Cambridge, MA, USA: Cambridge University Press.
- Shaosheng, F., & Hui, X. (2004). Diagonal recurrent neural network based predictive control for active power filter. In *Proceedings of the International Conference on Power Systems Technology, POWERCON 2004*, Singapore (pp. 759-762).
- Specht, D. F. (1988). Probabilistic neural networks for classification, mapping, or associative memory. In *Proceedings of the IEEE Conference on Neural Networks*, 1, 525–532. doi:10.1109/ICNN.1988.23887
- Specht, D. F. (1990). Probabilistic neural networks. *Neural Networks*, 3(1), 109–118. doi:10.1016/0893-6080(90)90049-Q
- Strukov, D. B., Snider, G. S., Stewart, D. R., & Williams, R. S. (2008, May). The missing memristor found. *Nature*, 453, 80–83. doi:10.1038/nature06932
- Thomas, T. J. (2000). *Locally-connected neural network architecture for invariant pattern recognition* (NPO-20633). USA: NASA Technical Briefings.
- Tsoi, A. C., & Back, A. D. (1994). Locally recurrent globally feedforward networks: A critical review of architectures. *IEEE Transactions on Neural Networks*, 5(2), 229–239. doi:10.1109/72.279187
- Tsoi, A. C., & Back, A. D. (1997). Discrete time recurrent neural network architectures: A unifying review. *Neurocomputing*, 15(3-4), 183–223. doi:10.1016/S0925-2312(97)00161-6
- Uchiyama, T., Shimohara, K., & Tokunaga, Y. (1989). A modified leaky integrator network for temporal pattern recognition. In *Proceedings of the International Joint Conference on Neural Networks*, 1, 469–475. doi:10.1109/IJCNN.1989.118621
- Uncini, A. (2002). Sound synthesis by flexible activation function recurrent neural networks. In M. Marinaro & R. Tagliaferri (Eds.), *Proceedings of the WIRN VIETRI 2002* (LNCS 2486, pp. 168-177).
- de Vries, B., & Principe, J. (1992). The gamma model – a new neural model for temporal processing. *Neural Networks*, 5(4), 565–576. doi:10.1016/S0893-6080(05)80035-8

Wang, X., & Peng, G. (2003). Modeling and control for pneumatic manipulator based on dynamic neural network. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* (Vol. 3, pp. 2231-2236).

Wang, X., Peng, G., & Xue, Y. (2003). Internal model controller with diagonal recurrent neural network for pneumatic robot servo system. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automatics*, Japan (pp. 1064-1069).

Yang, Z. R., & Chen, S. (1998). Robust maximum likelihood training of heteroscedastic probabilistic neural networks. *Neural Networks*, 11(4), 739–748. doi:10.1016/S0893-6080(98)00024-0

Yu, D. L., & Chang, T. K. (2005). Adaptation of diagonal recurrent neural network model. *Neural Computing & Applications*, 14, 189–197. doi:10.1007/s00521-004-0453-9

Zemouri, R., Racoceanu, D., & Zerhouni, N. (2003). Recurrent radial basis function network for time series prediction. *Engineering Applications of Artificial Intelligence*, 16, 453–463. doi:10.1016/S0952-1976(03)00063-0

KEY TERMS AND DEFINITIONS

Dynamic Neural Networks: Networks that incorporate dynamic synaptic or feedback weights among some or all of their neurons. These networks are capable of expressing dynamic behaviors.

Recurrent Neural Networks: Architectures that incorporate feedback connections among the layers of the network, or those that do not have straightforward layered input-output architecture but instead the inputs flow forth and back among the nodes of the network.

Locally Connected Recurrent Neural Networks: Feedforward architectures that incorporate a layer of recurrent neurons, which possess feedbacks only from neurons belonging to the same layer.

Non-Local Recurrent: Non-local recurrent structure is another term for denotation of the *locally connected recurrent* architectures.

Locally Recurrent Neural Networks: Networks that possess neurons with local feedback connections. On the level of individual neurons, local recurrence is identified with the availability of one or more feedbacks that encompass one or more elements in the structure of a neuron.

Chapter 10

Nonstationary Signal Analysis with Kernel Machines

Paul Honeine

Institut Charles Delaunay, France

Cédric Richard

Institut Charles Delaunay, France

Patrick Flandrin

Ecole Normale Supérieure de Lyon, France

ABSTRACT

This chapter introduces machine learning for nonstationary signal analysis and classification. It argues that machine learning based on the theory of reproducing kernels can be extended to nonstationary signal analysis and classification. The authors show that some specific reproducing kernels allow pattern recognition algorithm to operate in the time-frequency domain. Furthermore, the authors study the selection of the reproducing kernel for a nonstationary signal classification problem. For this purpose, the kernel-target alignment as a selection criterion is investigated, yielding the optimal time-frequency representation for a given classification problem. These links offer new perspectives in the field of nonstationary signal analysis, which can benefit from recent developments of statistical learning theory and pattern recognition.

INTRODUCTION

Time-frequency and time-scale distributions have become increasingly popular tools for analysis and processing of nonstationary signals. These tools map a one-dimensional signal into a two-dimensional distribution, a function of both time and frequency. Such joint description reveals the time-varying frequency content of nonstationary signals, unlike classical spectral analysis techniques *a la Fourier*. Over the years, a large variety of classes of time-frequency distributions have been proposed to explain the diversity of the treated problems. Linear and quadratic distributions have been extensively studied, and among them Cohen's class of time-frequency distributions as (quasi) energy distribution jointly

DOI: 10.4018/978-1-60566-766-9.ch010

in time and frequency, and most notably the Wigner-Ville distribution, see for instance (Cohen, 1989; Flandrin, 1999; Auger & Hlawatsch, 2008). From these, one can choose the optimal representation for the problem under investigation, such as increasing the representation immunity to noise and interference components (Auger & Flandrin, 1995, Baraniuk & Jones, 1993), or selecting the best class of representations for a given decision problem (Heitz, 1995; Till & Rudolph, 2000; Davy, Doncarly, & Boudreux-Bartels, 2001).

Over the last decade, multiple analysis and classification algorithms based on the theory of reproducing kernel Hilbert space (RKHS) have gained wide popularity. These techniques take advantage of the so-called *kernel trick*, which allows construction of nonlinear techniques based on linear ones. Initiated by state-of-the-art support vector machines (SVM) for classification and regression (Vapnik, 1995), the most popular ones include the nonlinear generalization of principal component analysis or kernel-PCA (Schölkopf, Smola, & Müller, 1998), and nonlinear Fisher's discriminant analysis or kernel-FDA (Mika, Rätsch, Weston, Schölkopf, & Müller, 1999); see (Shawe-Taylor & Cristianini, 2004) for a survey of kernel machines. Kernel machines are computationally attractive, with outstanding performance, validated theoretically by the statistical learning theory (Vapnik, 1995; Cucker & Smale, 2002). Despite these advances, nonstationary signal analysis and classification still has not benefited from these developments, although such techniques have been brought to the attention of the signal processing community. Few work combine kernel machines and time-frequency analysis, of these Davy *et al.* (2002) apply the SVM algorithm for classification with a reproducing kernel expressed in the time-frequency domain. More recently, Honeine *et al.* (2007) applied a large panel of kernel machines for nonstationary signal analysis and classification, while in Honeine *et al.* (2006) and in Honeine and Richard (2007) the optimality of the representation space is treated.

This chapter shows how the most effective and innovative kernel machines can be configured, with a proper choice of reproducing kernel, to operate in the time-frequency domain. Further, this approach is extended to the selection of the optimal time-frequency domain for a given classification task. For this purpose, the strength of the kernel-target alignment criterion is investigated for time-frequency distributions. The performance of the proposed approach is illustrated with simulation results. But before, a brief review of the principal elements of the theory behind kernel machines is presented.

RKHS AND KERNEL MACHINES: A BRIEF REVIEW

The theory behind RKHS serves as a foundation of the kernel machines. The main building blocks of these statistical learning algorithms are the *kernel trick* and the Representer Theorem. In this section, these concepts are presented succinctly, after a short introduction on reproducing kernels.

Reproducing Kernels and RKHS

Let X be a subspace of $L_2(\mathbb{C})$ the space of finite-energy complex signals, equipped with the usual inner product defined by $\langle x_i, x_j \rangle = \int_t x_i(t) x_j^*(t) dt$ and its corresponding norm, where $x_j^*(t)$ denotes the complex conjugate of the signal $x_j(t)$. A kernel is a function $\kappa(x_i, x_j)$ from $X \times X$ to \mathbb{C} , with Hermitian symmetry. The basic concept of reproducing kernels is described by the following two definitions (Aronszajn, 1950).

Definition 1. A kernel $\kappa(x_i, x_j)$ is said to be positive definite on X if the following is true:

$$\sum_{i=1}^n \sum_{j=1}^n a_i a_j^* \kappa(x_i, x_j) \geq 0 \quad (1)$$

for all $n \in \mathbb{N}$, $x_1, \dots, x_n \in X$ and $a_1, \dots, a_n \in C$.

Definition 2. Let $(H, \langle \cdot, \cdot \rangle_H)$ be a Hilbert space of functions from X to C . The function $\kappa(x_i, x_j)$ from $X \times X$ to C is the reproducing kernel of H if, and only if,

1. the function $\kappa_{x_j} : x_i \mapsto \kappa_{x_j}(x_i) = \kappa(x_i, x_j)$ belongs to H , for all $x_j \in X$;
2. one has $\psi(x_j) = \langle \psi, \kappa_{x_j} \rangle_H$ for all $x_j \in X$ and $\psi \in H$.

It can be shown that every positive definite kernel κ is the reproducing kernel of a Hilbert space of functions from X to C . It suffices to consider the space H_0 induced by the functions $\{\kappa_x\}_{x \in X}$, and equip it with the inner product

$$\langle \psi, \phi \rangle_{H_0} = \sum_{i=1}^n \sum_{j=1}^m a_i b_j^* \kappa(x_i, x_j) \quad (2)$$

where $\psi = \sum_{i=1}^n a_i \kappa_{x_i}$ and $\phi = \sum_{j=1}^m b_j \kappa_{x_j}$ are elements of H_0 . Then, this incomplete Hilbertian space is completed according to (Aronszajn, 1950), so that every Cauchy sequence converges in that space. Thus, one obtains the Hilbert space H induced by the reproducing kernel κ , usually called a *reproducing kernel Hilbert space*. It can also be shown that every reproducing kernel is positive definite (Aronszajn, 1950). A classic example of a reproducing kernel is the Gaussian kernel $\kappa(x_i, x_j) = \exp(-\|x_i - x_j\|^2/2\sigma^2)$, where σ is a tunable parameter corresponding to the kernel bandwidth. Other examples of reproducing kernels, and rules for designing and combining them can be found in (Vapnik, 1995; Shawe-Taylor & Cristianini, 2004).

The Kernel Trick, the Representer Theorem

Substituting ψ by κ_{x_i} in item 2 of Definition 2, one gets the following fundamental property of RKHS

$$\kappa(x_i, x_j) = \langle \kappa_{x_i}, \kappa_{x_j} \rangle_H \quad (3)$$

for all $x_i, x_j \in X$. Therefore, $\kappa(x_i, x_j)$ gives the inner product in H , the so-called *feature space*, of the images κ_{x_i} and κ_{x_j} of any pair of input data x_i and x_j , without having to evaluate them explicitly. This principle is called the *kernel trick*. It can be used to transform any linear data processing technique into

a nonlinear one, on the condition that the algorithm can be expressed in terms of inner products only, involving pairs of the input data. This is achieved by substituting each inner product $\langle x_i, x_j \rangle$ by a nonlinear kernel $\kappa(x_i, x_j)$, leaving the algorithm unchanged and incurring essentially the same computational cost.

In conjunction with the kernel trick, the Representer Theorem is a solid foundation of kernel machines for pattern recognition, such as SVM, kernel-PCA, and kernel-FDA (Schölkopf, Herbrich, & Smola, 2001).

Theorem (Representer Theorem). Any function φ of H minimizing a regularized cost function of the form

$$J((x_1, y_1, \phi(x_1)), \dots, (x_n, y_n, \phi(x_n))) + \rho(\|\phi\|_H^2) \quad (4)$$

with ρ a strictly monotonic increasing function on \mathbb{R}_+ , can be written as a kernel expansion in terms of the available data, namely

$$\phi(\cdot) = \sum_{j=1}^n a_j \kappa(\cdot, x_j). \quad (5)$$

Sketch of proof. Any function φ of the space H can be decomposed as $\varphi(\cdot) = \sum_{j=1}^n a_j \kappa(\cdot, x_j) + \phi_\perp(\cdot)$, where $\langle \phi_\perp(\cdot), \kappa(\cdot, x_j) \rangle_H = 0$ for all $j = 1, \dots, n$. Using this with (3), it is obvious that ϕ_\perp does not affect the value of $\varphi(x_i)$, for all $i = 1, \dots, n$. Moreover, one can verify that the n -order model defined in (5) minimizes ρ since $\rho(\|\varphi\|_H^2) = \rho(\|\phi\|_H^2 + \|\phi_\perp\|_H^2) \geq \rho(\|\phi\|_H^2)$ with equality only if $\varphi = \phi$. This is the essence of the Representer Theorem. \square

TIME-FREQUENCY KERNEL MACHINES: THE WIGNER-VILLE DISTRIBUTION

In this section, we investigate the use of kernel learning machines for pattern recognition in the time-frequency domain, by taking advantage of both the kernel trick and the Representer Theorem. To clarify the discussion, we will first focus on the Wigner-Ville distribution. This will be followed by an extension to other time-frequency distributions, linear and quadratic. Below, A_n denotes a training set of n instances $x_i \in X$ and the desired outputs or labels $y_i \in Y$, with $Y = \{\pm 1\}$ for a binary (2 classes) classification problem.

The Wigner-Ville Distribution

The Wigner-Ville distribution is considered fundamental among the large class of time-frequency representations. This is mainly due to many desirable theoretical properties such as the correct marginal conditions for instance, as well as the unitary condition. The latter propels this distribution into a suit-

able candidate for detection based on the time-frequency domain. The Wigner-Ville distribution of a finite energy signal $x(t)$ is given by

$$W_x(t, f) = \int x(t + \tau / 2) x^*(t - \tau / 2) e^{-2j\pi f\tau} d\tau. \quad (6)$$

Consider applying conventional linear pattern recognition algorithms directly to time-frequency representations. This means that one should optimize a given criterion J of the general form (4), where each signal x_i , for $i=1,2,\dots,n$, is substituted by its time-frequency distribution W_{x_i} . Therefore, one seeks to determine a function ϕ of the form

$$\phi(x) = \langle W_x, \Phi \rangle = \iint W_x(t, f) \Phi(t, f) dt df, \quad (7)$$

or equivalently the time-frequency pattern $\Phi(t, f)$. The main difficulty encountered in solving such problems is that they are typically very high dimensional, the size of the Wigner-Ville distributions calculated from the training set being quadratic in the length of signals, leading to manipulating n representations of size l^2 for signals of length l . This makes pattern recognition based on time-frequency representations time-consuming, if not impossible, even for reasonably-sized signals. With both the kernel trick and the Representer Theorem, kernel machines eliminate this computational burden. For this purpose, one may consider the inner product between the Wigner-Ville distributions, which is given by the kernel

$$\kappa_W(x_i, x_j) = \langle W_{x_i}, W_{x_j} \rangle. \quad (8)$$

Note however that the time-frequency distributions, W_{x_i} and W_{x_j} here, do not need to be computed in order to evaluate κ_W . For this purpose, one may consider the unitarity of the Wigner-Ville distribution illustrated by Moyal's formula $\langle W_{x_i}, W_{x_j} \rangle = |\langle x_i, x_j \rangle|^2$, yielding

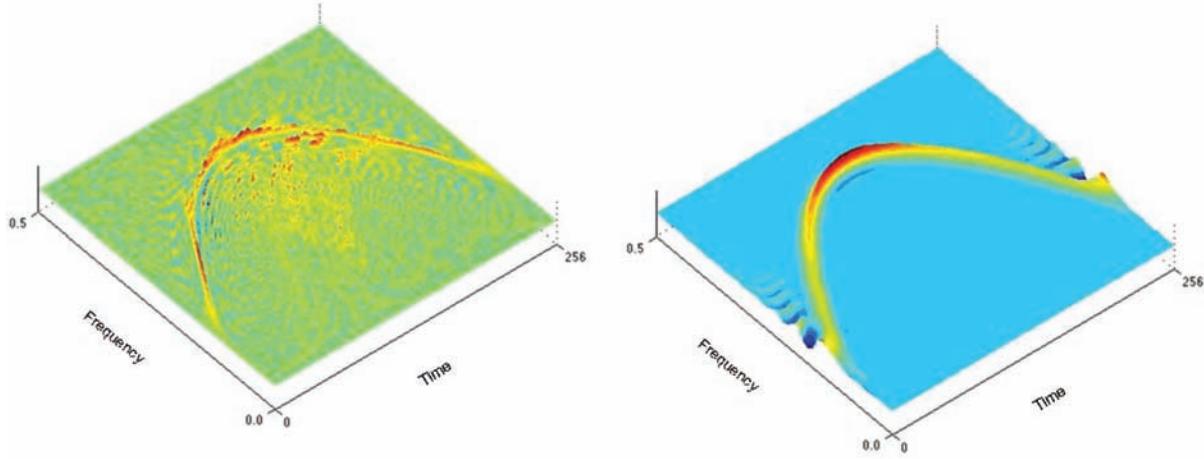
$$\kappa_W(x_i, x_j) = |\langle x_i, x_j \rangle|^2. \quad (9)$$

It is evident that κ_W is a positive definite kernel, since the condition given by Definition 1 is clearly verified as $\|\sum_j a_j W_{x_j}\|^2 \geq 0$. Therefore, we are now in a position to construct the RKHS induced by this kernel, and denoted by H_W . It is obtained by considering the space H_0 defined below, and complete it with the limit of every Cauchy sequence

$$H_0 = \{\phi : X \rightarrow \text{IR} \mid \phi(\cdot) = \sum_j a_j |\langle \cdot, x_j \rangle|^2, a_j \in \text{IR}, x_j \in X\}. \quad (10)$$

Thus, the kernel (9) can be considered with any kernel machine proposed in the literature to perform pattern recognition tasks in the time-frequency domain. By taking advantage of the Representer Theorem, the solution $\phi(\cdot) = \sum_j a_j |\langle \cdot, x_j \rangle|^2$ allows for a time-frequency distribution interpretation, since it can be written as $\phi(x) = \langle W_x, \Phi_W \rangle$, with the time frequency signature

Figure 1. First principal distribution resulting from the Wigner-Ville-based kernel-PCA (left), and the Born-Jordan-based kernel-PCA (right).



$$\Phi_W = \sum_{j=1}^n a_j W_{x_j}. \quad (11)$$

This expression is directly obtained by combining (5) and (7). One should keep in mind that the n coefficients a_j that determine the solution are estimated without calculating any Wigner-Ville distribution, since only their inner products are required. Subsequently, the time-frequency pattern Φ_W can be determined with (11) in an iterative manner, without suffering the drawback of storing and manipulating a large collection of Wigner-Ville distributions. Moreover, most of the kernel machines speed-up the calculation of the time-frequency pattern Φ_W since a large number of the resulting coefficients a_j is null. This sparsity of the solution made a breakthrough in the machine learning community with the highly-performant SVM algorithms.

Example of Time-Frequency Kernel Machine: The Wigner-Ville-Based PCA

In order to emphasize the main idea behind time-frequency kernel machines, the Wigner-Ville-based PCA approach is illustrated here, by considering the Wigner-Ville distribution and the kernel-PCA algorithm. The latter is a nonlinear form of PCA, and allows to extract principal components of variables that are nonlinearly related to the input variables. Given n centered observations x_1, \dots, x_n , a dual formulation of the standard PCA algorithm consists of diagonalizing the Gram matrix K whose (i,j) -th entry is $\langle x_i, x_j \rangle$. The i -th coordinate of the k -th principal component is then given by $\sum_{j=1}^n a_{j,k} \langle x_i, x_j \rangle$, where the $a_{j,k}$'s are the components of the k -th eigenvector of K . In kernel-PCA, a nonlinear transformation of the input data is carried out implicitly by replacing each inner product $\langle x_i, x_j \rangle$ with a kernel function $\kappa(x_i, x_j)$. A major interest of this method is that it performs PCA in feature spaces of arbitrarily large, possibly infinite, dimension. In particular, it can be adjusted to operate on the Wigner-Ville distributions of n signals x_1, \dots, x_n via an eigendecomposition of the Gram matrix K_W whose (i,j) -th entry is $\kappa_W(x_i, x_j)$. The k -th principal component can then be extracted from any signal x as follows

$$\phi_k(x) = \sum_{j=1}^n a_{j,k} \kappa_W(x, x_j), \quad (12)$$

where the $a_{j,k}$'s are the components of the k -th eigenvector of K_W . Now we can state an equivalent time-frequency formulation to the expression above: $\phi_k(x) = \langle W_x, \Phi_k \rangle$, with the signature

$$\Phi_k = \sum_{j=1}^n a_{j,k} W_{x_j}. \quad (13)$$

We call Φ_k the k -th principal distribution, although it may not be a valid time-frequency distribution.

To illustrate kernel-PCA as a potentially useful tool in nonstationary signal processing, a collection of 500 signals of 256 samples each is considered. Each signal consists of a quadratic frequency modulation between 0.1 and 0.4 in normalized frequency, corrupted with an additive white Gaussian noise at a signal-to-noise ratio of 0 dB. Figure 1 (left) shows the first principal distribution obtained by applying the kernel-PCA based on the Wigner-Ville time-frequency distribution. This is done by applying the classical kernel-PCA technique with the Wigner-Ville kernel (9), and then injecting the resulting weighting coefficients into (13). As illustrated in the figure, the quadratic frequency modulation and interference terms can be clearly identified in Φ_1 . In order to reduce interference components in the resulting principal distribution, one may consider a different time-frequency distribution, as illustrated in Figure 1 (right) with Born-Jordan distribution. In the following section, we extend the proposed approach to other time-frequency distributions of Cohen's class, whose elements include the Wigner-Ville distribution, the Born-Jordan distribution and the Choi-Williams distribution, only to name a few.

But before it is worth noting the gain in computational complexity by the proposed approach. Applying standard PCA directly to the set of Wigner-Ville distributions would lead to the same result. However, this approach suffers from the high computational cost of calculating, storing and manipulating a set of 500 matrices, each having size 256^2 by 256^2 . In addition, it requires calculating and diagonalizing a 256^2 by 256^2 covariance matrix, which is computationally intensive if not impossible. This conclusion remains valid for standard pattern recognition machines such as FDA and GDA.

EXTENSION TO OTHER TIME-FREQUENCY DISTRIBUTIONS

Obviously, the concept of time-frequency machines for pattern recognition is not limited only to the Wigner-Ville distribution. This section illustrates it with other popular time-frequency distributions, linear and quadratic. More particularly, the choice of the optimal representation for a given classification task is studied in next section, with the kernel-target alignment criterion.

Linear Representations

The short-time Fourier transform (STFT) remains the most widely used linear time-frequency distribution. For a given analysis window $w(t)$, well localized around the origin of the time-frequency domain, it is defined for a signal x by

$$F_x(t, f) = \int x(\tau) w^*(\tau - t) e^{-2j\pi f\tau} d\tau, \quad (14)$$

or, in an equivalent way, $F_x(t, f) = \langle x, w_{tf} \rangle$ with $w_{tf}(\tau) = w(\tau - t) e^{2j\pi f\tau}$. We now consider the following kernel function $\kappa_F(x_i, x_j) = \langle F_{x_i}, F_{x_j} \rangle$, namely,

$$\kappa_F(x_i, x_j) = \|w\|^2 \langle x_i, x_j \rangle. \quad (15)$$

It is obvious that this is a positive definite kernel, since condition (1) takes the form $\|\sum_i a_i x_i\|^2 \geq 0$ which is trivially satisfied. Therefore, kernel κ_F induces a RKHS and can be used with any kernel machine to operate on the short-time Fourier domain. From the Representer Theorem, equation (5) offers a time-frequency distribution interpretation, as $\varphi(x) = \langle F_x, \Phi_F \rangle$ with the time-frequency signature $\Phi_F = \sum_{j=1}^n a_j F_{x_j}$. Moreover, since the considered representation is linear with respect to the signal, this is equivalent to $\Phi_F = F_z$ with z being the signal $z = \sum_{j=1}^n a_j x_j$. Therefore, one needs to evaluate the short-time Fourier transform only once.

This illustrates the potential use of any kernel machine for pattern recognition with the short-time Fourier representation. Obviously, this is not limited only to this linear representation. For instance, one may consider the Fourier transform, defined by $\hat{x}(f) = \int x(\tau) e^{-2j\pi f\tau} d\tau$ for a signal x . In a more general case, one may also consider a time-scale representation, such as the wavelet transform as illustrated next.

The (continuous) wavelet decomposition is a very appreciated linear representation, relying on a time-translation of t , and a scaling factor a of a mother wavelet w , such as $w_{t,a}(\tau) = a^{-1/2} w((\tau - t)/a)$. The wavelet representation of a given signal x can be written as

$$\Omega_x(t, a) = \int x(\tau) \frac{1}{\sqrt{a}} w^*((\tau - t)/a) d\tau.$$

In order to be an invertible transformation, the admissible condition $c_w = \int |\hat{x}(f)|^2 df / \|f\| < +\infty$ must be satisfied. This not-too restrictive condition on the mother wavelet is verified for instance by the mexican hat, which is given by the second derivative of the Gaussian. This admissibility condition allows an identity similar to (15) with

$$\iint \Omega_{x_i}(t, a) \Omega_{x_j}^*(t, a) \frac{dt da}{a^2} = c_w \int x_i(t) x_j^*(t) dt$$

where $dt da / a^2$ is a natural measure associated to the translation and scaling (Flandrin, 1999). Therefore, by normalizing the mother wavelet such that $c_w = 1$, we get a unitary transformation.

To summarize, both the short-time Fourier transform and the wavelet transform, as well as the Fourier transform, are linear with respect to the studied signal. Therefore, they share the same reproducing kernel defined by the linear kernel $\kappa(x_i, x_j) = \langle x_i, x_j \rangle$, upto a multiplicative normalization constant. Next, we extend this approach to the quadratic class of time-frequency distributions, as defined by Cohen's class.

Quadratic Time-Frequency Distributions

The concept of quadratic forms for nonstationary signal analysis is mainly driven by the need to study its energy distribution over both time and frequency, simultaneously. This has produced a large body of theoretical work over the years, as many different classes of solutions emerge naturally. From these, the Cohen class of time-frequency distributions gained considerable attention in recent years. These distributions are covariant with respect to time-frequency shifts applied to the signal under scrutiny. For a finite energy signal $x(t)$ to be analyzed, a distribution belonging to Cohen class distributions is given by

$$C_x(t, f) = \iint \Pi(t' - t, f' - f) W_x(t', f') dt' df' \quad (16)$$

where W_x is the Wigner-Ville distribution of the signal x and Π is a two-dimensional weighting function. The latter determines the properties of the distribution. We can easily check that $\kappa_\Pi(x_i, x_j) = \langle C_{x_i}, C_{x_j} \rangle$ is a positive definite kernel. Then it can be used by any kernel machine, leading to the solution $\phi(x) = \langle C_x, \Phi_\Pi \rangle$, with $\Phi_\Pi = \sum_{j=1}^n a_j C_{x_j}$. The Π -tunable kernel κ_Π provides a large class of solutions adapted for a given task. For instance, one may require solutions that are relatively immune to interference and noise for analysis purpose. The kernel can also be exploited to improve classification accuracy, by maximizing a contrast criterion between classes.

The spectrogram is probably the most popular distribution of the Cohen class. Defined as the squared magnitude of the short-time Fourier transform (14), it is related to the kernel $\kappa_s(x_i, x_j) = \iint |\langle x_i, w_{t,f} \rangle \langle x_j, w_{t,f} \rangle|^2 dt df$. Other examples of the Cohen class include distributions verifying the unitary condition, such as the Wigner-Ville distribution, the Rihaczek distribution and the Page distribution. While these distributions share the same kernel (9), a property resulting from Moyal's formula $\langle C_{x_i}, C_{x_j} \rangle = |\langle x_i, x_j \rangle|^2$, they differ by the time-frequency pattern Φ_C . The latter can be computed directly of using Φ_W with

$$\Phi_\Pi(t, f) = \iint \Pi(t' - t, f' - f) \Phi_W(t', f') dt' df' \quad (17)$$

Examples of the most used time-frequency distributions of Cohen's class are presented in Table 1, with their definitions.

Without loss of generality, we illustrate this extension to a particular distribution of Cohen's class, the Born-Jordan distribution. Figure 1 (right) shows the first principal distribution obtained by considering the kernel-PCA algorithm with the reproducing kernel associated to the Born-Jordan distribution, for the same data as in Figure 1 (left) for the Wigner-Ville distribution. We recall that this time-frequency distribution is defined for a given signal $x(t)$ by

$$BJ_x(t, f) = \int \frac{1}{|\tau|} \int_{t-|\tau|/2}^{t+|\tau|/2} x(t' + \tau/2) x^*(t' - \tau/2) dt' e^{-2j\pi f\tau} d\tau,$$

leading to a reduced-interference distribution (Flandrin, 1999). This property is inherited into the principal distribution as illustrated in Figure 1 (right). Table 2 summarizes kernels associated to some time-frequency distributions, linear and quadratic, as illustrated all over this chapter.

While computing the kernel $\kappa_w(x_i, x_j)$ is efficient as illustrated for the Wigner-Ville-based PCA, this is not the case for any arbitrary non-unitary distribution, leading to a time consuming process for training a kernel machine. In applications where computation time is a major issue, a simple heuristic procedure can be considered to derive solutions of the form $\phi(x) = \langle C_x, \Phi_{\Pi} \rangle$. For this purpose, the kernel machine is trained by considering the easy-to-compute Wigner-Ville kernel $\kappa_w(x_i, x_j) = |\langle x_i, x_j \rangle|^2$. By combining the distribution Φ_w resulting from (11) with equation (17) we get the time-frequency feature Φ_C . Clearly, this is a non-optimal strategy when the considered distribution does not satisfy the unitary condition.

OPTIMAL TIME-FREQUENCY REPRESENTATION: THE KERNEL-TARGET ALIGNMENT CRITERION

In previous sections, we have shown how kernel machines can be configured to operate in the time-frequency domain, with the proper choice of reproducing kernel. This section is devoted to the question of selecting the appropriate time-frequency distribution, and therefore the associated reproducing kernel, for a given classification task. Next, the kernel-target alignment criterion is studied for selecting a time-frequency representation, after this succinct review on different available strategies for kernel selection.

Table 1.

Wigner-Ville	$\int x(t + \tau / 2) x^*(t - \tau / 2) e^{-2j\pi f\tau} d\tau$
Born-Jordan	$\int \frac{1}{ \tau } \int_{t-\tau/2}^{t+\tau/2} x(t' + \tau / 2) x^*(t' - \tau / 2) dt' e^{-2j\pi f\tau} d\tau$
Rihaczek	$x(t) \hat{x}^*(f) e^{-2j\pi ft}$
Margeneau-Hill	$\text{Re}\{x(t) \hat{x}^*(f) e^{-2j\pi ft}\}$
Choi-Williams	$\iint \frac{\sigma}{ \tau } e^{-2\sigma^2(t'-t)^2/\tau^2} x(t' + \tau / 2) x^*(t' - \tau / 2) e^{-2j\pi f\tau} dt' d\tau$
Butterworth	$\iint \frac{\sqrt{\sigma}}{2 \tau } e^{- t' \sigma^2 / \tau } x(t + t' + \tau / 2) x^*(t + t' - \tau / 2) e^{-2j\pi f\tau} dt' d\tau$
Spectrogram	$\left \int x(\tau) w^*(\tau - t) e^{-2j\pi f\tau} d\tau \right ^2$

Kernel Selection in Machine Learning: A Brief Review

Many results from the statistical learning theory emphasize on the crucial role of prior knowledge in machine learning problems. For the large class of kernel machines, the *no free kernel theorem* (Cristianini, Kandola, Elisseeff, & Shawe-Taylor, 2006) shows that no kernel is optimal for all applications, and therefore any prior knowledge must contribute to the choice of the appropriate kernel. For this purpose, many algorithms have been proposed in order to reveal the relationship between the data and their labels.

Cross-validation strategies, as well as *leave-one-out* techniques, are widely used in the literature to evaluate the performance for the model. For a given kernel, the generalization error is estimated by constructing several classifiers from various subsets of the available data, and then validated on the remaining data (Meyer, Leisch, & Hornik, 2003). This strategy is conducted on several candidate kernels, and the optimal kernel corresponds to the one with the smallest estimated error. For a tunable kernel, the optimal value of the tuning parameter is obtained from a grid search. The cross-validation approach turns out to be highly time consuming, since it requires a large number of train-and-validate stages. In order to speed up calculations, schemes to estimate a bound on the generalisation error have been recently proposed, requiring only one classifier per candidate kernel. From these, we recall the VC-dimension bound (Burges, 1998), the radius-margin criterion (Chung, Kao, Sun, Wang, & Lin, 2003), and the generalized approximate cross validation (Wahba, Lin, & Zhang, 2000). Other schemes are studied for instance in (Chapelle, Vapnik, Bousquet, & Mukherjee, 2002) (Duan, Keerthi, & Poo, 2003). While the framework defined by these methods is motivated from a theoretical point of view, their computational complexity is often cumbersome.

It is worth noting that these methods can be easily adapted to the time-frequency domain, as well as their statistical and theoretical properties. Next, a reduced computational complexity criterion is considered,

Table 2. Some kernels associated to linear and quadratic time-frequency distributions.

	Distribution name	Associated reproducing kernel
linear	Fourier	$\kappa(x_i, x_j) = \langle x_i, x_j \rangle$
	Short-time Fourier (STFT)	$\kappa_F(x_i, x_j) = \ w\ ^2 \langle x_i, x_j \rangle$
	Wavelet	$\kappa_{wav}(x_i, x_j) = c_w \langle x_i, x_j \rangle$
quadratic	Wigner-Ville	$\kappa_W(x_i, x_j) = \langle x_i, x_j \rangle ^2$
	Spectrogram	$\kappa_S(x_i, x_j) = \iint \langle x_i, w_{t,f} \rangle \langle x_j, w_{t,f} \rangle ^2 dt df$
	Page, Rihaczek	$\kappa_W(x_i, x_j) = \langle x_i, x_j \rangle ^2$
	Cohen	$\kappa_\Pi(x_i, x_j) = \langle C_{x_i}, C_{x_j} \rangle$

the kernel-target alignment, and its adaptation for selecting time-frequency distributions is studied.

The Kernel-Target Alignment Criterion

Performance of kernel-based pattern recognition methods is essentially influenced by the considered reproducing kernel. For a given algorithm such as the SVM, results obtained from two different kernels are as close as these kernels are similar. For this purpose, the authors of (Cristianini, Shawe-Taylor, Elisseeff, & Kandola, 2001) introduced the alignment as a measure of similarity between two reproducing kernels. Given a learning set $A_n = \{(x_1, y_1), \dots, (x_n, y_n)\}$ of n data and their labels, the empirical alignment between two kernels κ_1 and κ_2 is defined by

$$A(\kappa_1, \kappa_2, A_n) = \frac{\langle K_1, K_2 \rangle_F}{\sqrt{\langle K_1, K_1 \rangle_F \langle K_2, K_2 \rangle_F}}, \quad (18)$$

where K_1 and K_2 the Gram matrices with entries of the form $\kappa_1(x_i, x_j)$ and $\kappa_2(x_i, x_j)$ respectively, and $\langle \cdot, \cdot \rangle_F$ denotes Frobenius inner product defined by $\langle K_1, K_2 \rangle_F = \sum_i \sum_j \kappa_1(x_i, x_j) \kappa_2(x_i, x_j)$. The alignment corresponds to a correlation coefficient between both matrices K_1 and K_2 .

Recall that for a classification problem, one seeks a decision rule $\phi^*(\cdot)$ satisfying the relation $\phi^*(x_i) = y_i$ for $i = 1, \dots, n$. For the particular case of a 2-class problem, this relation takes the form $\phi^*(x_i) = \pm 1$, depending on which class x_i belongs. The reproducing kernel corresponding to this ideal transformation can be defined by $\kappa^*(x_i, x_j) = y_i y_j$. The associated ideal Gram matrix $K^* = \mathbf{y} \mathbf{y}'$, where \mathbf{y} denotes the target column vector whose i -th entry is y_i , which leads to

$$K^*(i, j) = \begin{cases} 1 & \text{si } y_i = y_j \\ -1 & \text{si } y_i \neq y_j \end{cases} \quad (19)$$

In what follows, the 2-class classification problem is studied, with K^* as defined above. But before, we emphasize on the simplicity of generalizing the proposed approach for a multi-class problem. For the case of c classes, the targets correspond to the c unit-norm and equiangular vectors, leading to $\kappa^*(x_i, x_j) = 1/(1 - c)$ if x_i and x_j belong to different classes, and $\kappa^*(x_i, x_j) = 1$ otherwise.

In (Cristianini, Shawe-Taylor, Elisseeff, & Kandola, 2001 ; Cristianini, Kandola, Elisseeff, & Shawe-Taylor, 2006), Cristianini *et al.* suggest to use the alignment to measure the similarity between a given reproducing kernel and the ideal target matrix $K^* = \mathbf{y} \mathbf{y}'$, in order to determine its relevance for the classification task in hand. Therefore, one considers the an optimization problem of the form

$$\kappa^* = \arg \max_{\kappa} \frac{\sum_{i,j=1}^n y_i y_j \kappa(x_i, x_j)}{n(\sum_{i,j=1}^n (\kappa(x_i, x_j))^2)^{1/2}}$$

where the definition of the alignment (18) is considered with $K^* = \mathbf{y} \mathbf{y}'$. The relevance of the alignment criterion is provided by a connection to the error of generalization, as demonstrated in (Cristianini,

Shawe-Taylor, Elisseeff, & Kandola, 2001) on a Parzen estimator of the form $g(\cdot) = \frac{1}{n} \sum_{i=1}^n a_i \kappa(\cdot, x_i)$. Since the first study of Cristianini *et al.*, the concept of maximizing the kernel-target alignment has been extended to other problems, including regression problems (Kandola, Shawe-Taylor, & Cristianini, On the extensions of kernel alignment, 2002), metric learning (Wu, Chang, & Panda, 2005 ; Lanckriet, Cristianini, Bartlett, Ghaoui, & Jordan, 2002), as well as studying the optimal combination of kernels in order to increase the performance of the classifier (Kandola, Shawe-Taylor, & Cristianini, 2002).

Optimal Time-Frequency Distribution

One should emphasize that the kernel-target alignment criterion does not require any learning of the decision rule. For this purpose, we investigate this approach to optimize a time-frequency representation for a classification task involving nonstationary signals. In a decisional framework, conventional strategies for determining the proper time-frequency representation mainly consist in selecting the one which yields the smallest estimated classification error (Heitz, 1995 ; Atlas, Droppo, & McLaughlin, 1997 ; Davy & Doncarli, 1998). However, this empirical approach requires multiple phases of learning the rule and cross-validating it. As we have seen so far, it is obvious that kernel machines for pattern recognition provide a unified context for solving a wide variety of statistical modelling and function estimation, for nonstationary signal analysis and classification.

The proposed approach involves kernels associated to time-frequency distributions, as defined in Table 2, and more particularly Cohen's class with the Π -tunable distributions. Therefore, the kernel-target alignment criterion can be written as

$$\frac{\sum_{i,j=1}^n y_i y_j \kappa_{\Pi}(x_i, x_j)}{n (\sum_{i,j=1}^n (\kappa_{\Pi}(x_i, x_j))^2)^{1/2}}, \quad (20)$$

and maximizing this score leads to the optimal time-frequency distribution, with optimality relative to the given classification task, defined by the available learning set of signals and their labels. The relevance of this approach is illustrated in next section, for nonstationary signals.

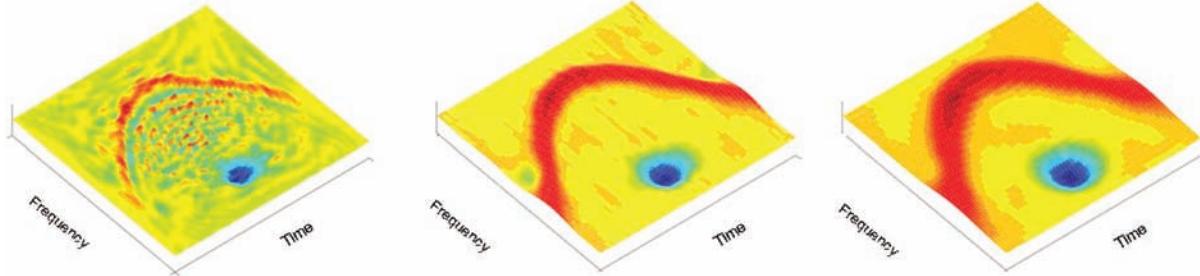
SIMULATION RESULTS

In this section, we study the relevance of the approach proposed in this chapter. Previously, we have illustrated the case of the kernel-PCA with the Wigner-Ville distribution. In Figure 1, we have shown the resulting principal time-frequency distributions for a conventional class of frequency modulated signals. In this section, we study learning a decisional problem for a classification/discrimination task, involving nonstationary signals.

Nonstationary Signal Classification: SVM and Kernel-FDA

This first set of experiments concerns a discrimination problem between two classes of 64-sample signals embedded in a white Gaussian noise of variance 1.25. The first class consists of 500 signals with

Figure 2. The resulting distributions obtained from the kernel-FDA algorithm with reproducing kernels associated to Wigner-Ville (left), the Choi-Williams (middle), and the spectrogram (right).

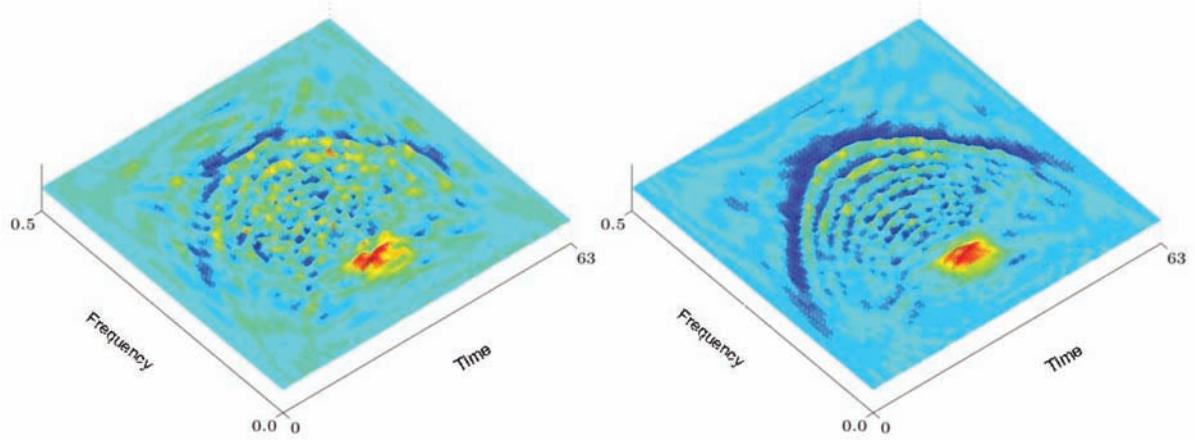


a quadratic frequency modulation between 0.1 and 0.4 in normalized frequency, and the second one of 500 signals with a Gaussian time-frequency atom in frequency at 0.1 and in time at the middle of the signals.

We apply the kernel-FDA algorithm with the quadratic kernel associated to the Wigner-Ville distribution. Figure 2 (left) presents the resulting distribution, where we get the two key components for the discrimination, on the one hand the quadratic frequency modulation with a positive orientation (red color) and on the other hand the Gaussian atom represented with a negative orientation (blue color). Moreover, some inference components are present in this signature, an inherent property of the Wigner-Ville distribution. In order to get rid of these interference components, one may consider other time-frequency distributions from Cohen's class, and use the associated reproducing kernel. For this purpose, we use both the Choi-Williams time-frequency distribution and the spectrogram, with their respective reproducing kernels. The resulting time-frequency signatures are illustrated in Figure 2 (middle) and Figure 2 (right), respectively. For each case, we found again the quadratic frequency modulation and the Gaussian atom, each in an opposed orientation. As opposed to the one obtained from the Wigner-Ville distribution, these signatures have reduced interferences. However, the price to pay is the relative high computational burden for evaluating the associated reproducing kernel. As illustrated in Figure 2, the proposed approach allows a time-frequency interpretation as opposed to the conventional linear techniques.

Moreover, we can study the relevance of several reproducing kernels for a given nonstationary signal classification problem, and using different learning algorithms. Two learning algorithms are investigated, the kernel-FDA and the classical SVM algorithms. The latter determines a separating hyperplan, with maximal-margin between the classes of the training set. This is mainly motivated by the statistical learning theory (Vapnik, 1995), and we wish to take advantage of this by applying it to the time-frequency domain, using an appropriate reproducing kernel as illustrated in this chapter. Therefore, we seek a time-frequency signature maximizing the distance with the time-frequency distributions of the training signals. For experiments, we consider the same discrimination problem as above, between quadratic frequency modulation signals and Gaussian atom signals. In order to compare the resulting estimated error, signals are corrupted further, with a signal-to-noise ratio of -10 dB. For now, we use the quadratic reproducing kernel associated to the Wigner-Ville distribution, and show in Figure 3 the resulting signatures for kernel-FDA and SVM algorithms. The latter shows better performance than the former, as illustrated by comparing Figure 3 (right and left), a property resulting from the inherent regularized property of the SVM.

Figure 3: Time-frequency distributions obtained with the quadratic Wigner-Ville reproducing kernel, using the kernel-FDA algorithm (left) and the classical SVM algorithm (right).



In order to compare the performance of several time-frequency distributions, we estimate the generalization error from a set of 10 000 signals belonging to both classes. We show in Table 3 the resulting estimated errors, even though the regularization parameter is not optimally tuned. The Wigner-Ville distribution is the best for the given classification task. The spectrogram and other *smooth* distributions are less adapted for this problem. This leads to the open question: how to choose the optimal distribution, without the computational burden of validating with a large set of test signals? This is accessible with the kernel-target alignment criterion, illustrated next for different time-frequency distributions.

Optimal Time-Frequency Distribution

Consider the classification of two families of signals of the form $e^{2j\pi\theta(t)}$ corrupted by an additive white Gaussian noise, where $\theta(t)$ is a linear phase modulation for the first class, increasing from 0.1 to 0.4 normalized frequencies, and quadratic for the second class, between .1 and .4 normalized frequencies. It is worth noting that signals from both classes share the same spectral support, which is not true in the time-frequency domain. For the learning task, consider 500 signals of length 256 for each family. On the one hand, the kernel-target alignment is computed from this training set, as given in expression (20) for different time-frequency distributions. On the other hand, these results are confronted with the

Table 3. Estimated generalization error (in %) for several time-frequency distributions, using the kernel-FDA and the SVM algorithms.

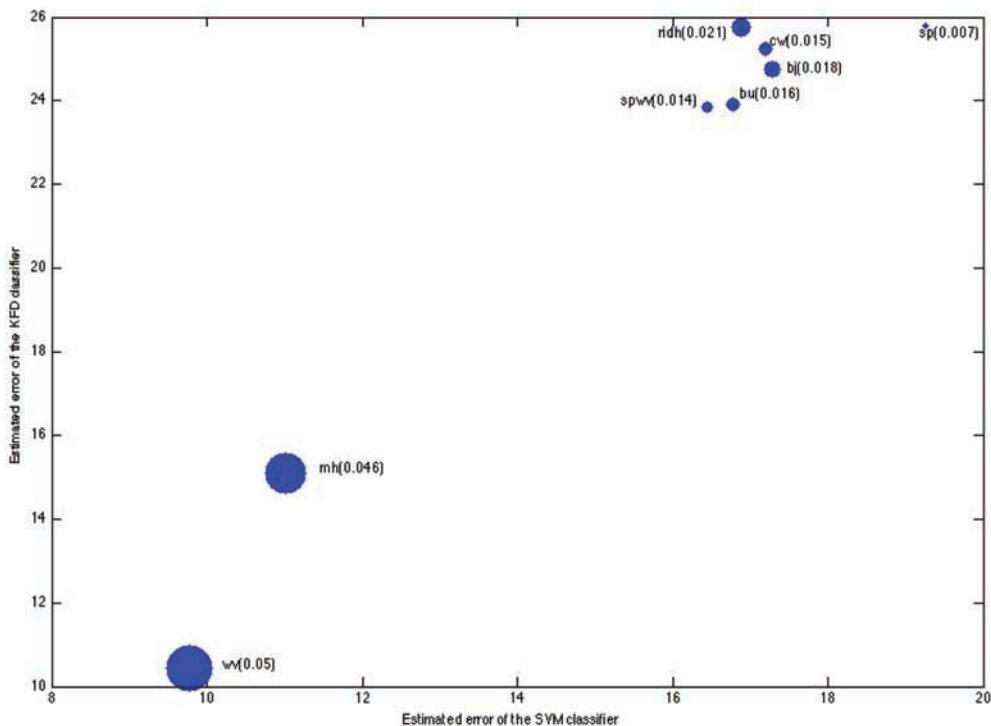
Time-frequency distribution	Kernel-FDA	SVM
Spectrogram	28.0	19.8
Smoothed Wigner-Ville	26.5	16.4
Choi-Williams	26.5	16.8
Wigner-Ville	19.3	13.5

performance of classifiers obtained from the same training set. For a given reproducing kernel, two classifiers are constructed, based independently on the SVM and on the kernel-FDA algorithms, and their generalization errors are estimated from a set of 20 000 signals. Figure 4 shows the performance of the kernels associated to the Wigner-Ville (wv), Margenau-Hill (mh), Choi-Williams (cw), Born-Jordan (bj), reduced interference with Hanning window (ridh), spectrogram (sp), and Butterworth (bu) distributions¹. It shows the relationship between the error rate for both classifiers and the kernel-target alignment score. Moreover, the ease with which the latter can be estimated using only training data, prior to any computationally intensive training, makes it an interesting tool for time-frequency distribution selection.

CONCLUSION

In this chapter, the authors presented the time-frequency kernel machines, a new framework for non-stationary signal analysis and processing. It is shown that pattern recognition algorithms based on the reproducing kernels can operate in the time-frequency domain, with some specific kernels. The choice of the proper kernel for a given classification task is studied from the kernel-target criterion, yielding new and efficient techniques for optimal time-frequency distribution selection. All these links offer new perspectives in the field of nonstationary signal analysis since they provide an access to the most recent developments of pattern recognition and statistical learning theory.

Figure 4. Error rate of a SVM classifier and a KFD classifier for different kernels associated to time-frequency distributions. The corresponding kernel-target alignment score is given between parentheses and determines the size of the dot.



REFERENCES

- Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68, 337–404. doi:10.2307/1990404
- Atlas, L., Droppo, J., & McLaughlin, J. (1997). Optimizing time-frequency distributions for automatic classification. *SPIE-The International Society for Optical Engineering*, 3162, 161–171.
- Auger, F., & Flandrin, P. (1995). Improving the readability of time-frequency and time-scale representations by the reassignment method. *IEEE Transactions on Signal Processing*, 43(5), 1068–1089. doi:10.1109/78.382394
- Auger, F., & Hlawatsch, F. (2008). *Time-frequency analysis*. New York: Wiley-ISTE.
- Baraniuk, R. G., & Jones, D. L. (1993). A signal-dependent time-frequency representation: Optimal kernel design. *IEEE Transactions on Signal Processing*, 41, 1589–1602. doi:10.1109/78.212733
- Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2), 121–167. doi:10.1023/A:1009715923555
- Chapelle, O., Vapnik, V., Bousquet, O., & Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1/3), 131–159. doi:10.1023/A:1012450327387
- Chung, K.-M., Kao, W.-C., Sun, C.-L., Wang, L.-L., & Lin, C.-J. (2003). Radius margin bounds for support vector machines with the RBF kernel. *Neural Computation*, 15(11), 2643–2681. doi:10.1162/089976603322385108
- Cohen, L. (1989). Time-frequency distributions-a review. *Proceedings of the IEEE*, 77(7), 941–981. doi:10.1109/5.30749
- Cristianini, N., Kandola, J., Elisseeff, A., & Shawe-Taylor, J. (2006). On kernel target alignment. In *Innovations in machine learning* (Vol. 194, pp. 205–256). Berlin, Germany: Springer.
- Cristianini, N., Shawe-Taylor, J., Elisseeff, A., & Kandola, J. (2001). On kernel-target alignment. In T. G. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Neural information processing systems (NIPS) 14* (pp. 367–373). Cambridge, MA: MIT Press.
- Cucker, F., & Smale, S. (2002). On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39(1), 1–49. doi:10.1090/S0273-0979-01-00923-5
- Davy, M., & Doncarli, C. (1998). Optimal kernels of time-frequency representations for signal classification. In *Proceedings of the IEEE International Symposium on TITS* (pp. 581–584).
- Davy, M., Doncarly, C., & Boudreux-Bartels, G. (2001). Improved optimization of time-frequency based signal classifiers. *IEEE Signal Processing Letters*, 8(2), 52–57. doi:10.1109/97.895373
- Davy, M., Gretton, A., Doucet, A., & Rayner, P. W. (2002). Optimised support vector machines for nonstationary signal classification. *IEEE Signal Processing Letters*, 9(12), 442–445. doi:10.1109/LSP.2002.806070

- Duan, K., Keerthi, S., & Poo, A. (2003). Evaluation of simple performance measures for tuning svm hyperparameters. *Neurocomputing*, 51, 41–59. doi:10.1016/S0925-2312(02)00601-X
- Flandrin, P. (1999). *Time-frequency/time-scale analysis*. San Diego, CA: Academic Press.
- Heitz, C. (1995). Optimum time-frequency representations for the classification and detection of signals. *Applied Signal Processing*, 3, 124–143.
- Honeine, P., & Richard, C. (2007). Signal-dependent time-frequency representations for classification using a radially gaussian kernel and the alignment criterion. In *Proc. of the IEEE Statistical Signal Processing Workshop*, Madison, WI, USA.
- Honeine, P., Richard, C., & Flandrin, P. (2007). Time-frequency kernel machines. *IEEE Transactions on Signal Processing*, 55, 3930–3936. doi:10.1109/TSP.2007.894252
- Honeine, P., Richard, C., Flandrin, P., & Pothin, J.-B. (2006). Optimal selection of time-frequency representations for signal classification: a kernel-target alignment approach. In *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Toulouse, France.
- Kandola, J., Shawe-Taylor, J., & Cristianini, N. (2002). *On the extensions of kernel alignment*. London: University of London, Department of Computer Science.
- Kandola, J., Shawe-Taylor, J., & Cristianini, N. (2002). *Optimizing kernel alignment over combinations of kernels*. London: University of London, Department of Computer Science.
- Lanckriet, G., Cristianini, N., Bartlett, P., Ghaoui, L. E., & Jordan, M. (2002). Learning the kernel matrix with semi-definite programming. In *Proc. of the 19th International Conference on Machine Learning* (pp. 323-330).
- Meyer, D., Leisch, F., & Hornik, K. (2003). The support vector machine under test. *Neurocomputing*, 55, 169–186. doi:10.1016/S0925-2312(03)00431-4
- Mika, S., Rätsch, G., Weston, J., Schölkopf, B., & Müller, K. R. (1999). Fisher discriminant analysis with kernels. In Y. H. Hu, J. Larsen, E. Wilson, & S. Douglas (Eds.), *Advances in neural networks for signal processing* (pp. 41-48). San Mateo, CA: Morgan Kaufmann.
- Schölkopf, B., Herbrich, R., & Smola, A. J. (2001). A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory and 5th European Conference on Computational Learning Theory (COLT '01/EuroCOLT '01)* (pp. 416-426). London, UK: Springer-Verlag.
- Schölkopf, B., Smola, A., & Müller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5), 1299–1319. doi:10.1162/089976698300017467
- Shawe-Taylor, J., & Cristianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge, UK: Cambridge University Press.
- Till, M., & Rudolph, S. (2000). Optimized time-frequency distributions for signal classification with feed-forward neural networks. In [Orlando: SPIE Proceedings Series.]. *Proceedings of the Applications and Science of Computational Intelligence III*, 4055, 299–310.
- Vapnik, V. (1995). *The nature of statistical learning theory*. New York: Springer-Verlag.

Wahba, G., Lin, Y., & Zhang, H. (2000). Generalized approximate cross validation for support vector machines. In A. Smola, P. Bartlett, B. Schölkopf, & C. Schuurmans (Eds.), *Advances in large margin classifiers* (pp. 297-309). Cambridge, MA: MIT Press.

Wu, G., Chang, E. Y., & Panda, N. (2005). Formulating distance functions via the kernel trick. In *Proceedings of the 11th ACM International Conference on Knowledge Discovery in Data Mining* (pp. 703-709).

KEY TERMS AND DEFINITIONS

Positive Definite Kernel: A two-variable function defined on X that satisfies $\sum_{i,j} a_i a_j^* \kappa(x_i, x_j) \geq 0$ for all $x_1, \dots, x_n \in X$ and $a_1, \dots, a_n \in C$.

Reproducing Kernel Hilbert Space: A Hilbert space of functions from X to C that possesses a reproducing kernel, i.e. a (positive definite) kernel $\kappa(x_i, x_j)$ with the properties: (1) $\kappa_x = \kappa(x, \cdot)$ belongs to that space, and (2) $\langle \psi, \kappa_x \rangle = \psi(x)$, for all $x \in X$ and $\psi \in H$.

Kernel-PCA: A nonlinear extension of the classical Principal Component Analysis algorithm based on the kernel paradigm, yielding a powerful feature extraction technique.

Kernel-Target Alignment Criterion: A criterion to select and tune a kernel for a given learning task, prior to any leaning, by comparing it to an ideal kernel obtained from the available training data.

Wigner-Ville Distribution: A High resolution joint time-frequency distribution for nonstationary signals analysis, defined by $W_x(t, f) = \int x(t + \tau / 2) x^*(t - \tau / 2) e^{-2j\pi f\tau} d\tau$ for a given signal x .

Cohen's Class of Time-Frequency Distributions: The class of distributions covariant with respect to time and frequency shifts applied to the studied signal. For a given signal x , a distribution belonging to Cohen's class is given by $C_x(t, f) = \iint \Pi(t' - t, f' - f) W_x(t', f') dt' df'$ where W_x is the Wigner-Ville distribution of x and Π is a tunable function.

Short-Time Fourier Transform: A linear time-frequency representation of a signal, defined by $F_x(t, f) = \int x(\tau) w^*(\tau - t) e^{-2j\pi f\tau} d\tau$ for a given analysis window w .

Wavelet Representation: A linear time-frequency representation relying on a time-translation and a scaling of a mother wavelet w , and defined by $\Omega_x(t, a) = \int x(\tau) a^{-1/2} w^*((\tau - t) / a) d\tau$.

ENDNOTE

¹ For tuning parameters, we consider those given by default in the Time-Frequency Toolbox.

Chapter 11

Transfer Learning

Lisa Torrey

University of Wisconsin, USA

Jude Shavlik

University of Wisconsin, USA

ABSTRACT

Transfer learning is the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned. While most machine learning algorithms are designed to address single tasks, the development of algorithms that facilitate transfer learning is a topic of ongoing interest in the machine-learning community. This chapter provides an introduction to the goals, settings, and challenges of transfer learning. It surveys current research in this area, giving an overview of the state of the art and outlining the open problems. The survey covers transfer in both inductive learning and reinforcement learning, and discusses the issues of negative transfer and task mapping.

INTRODUCTION

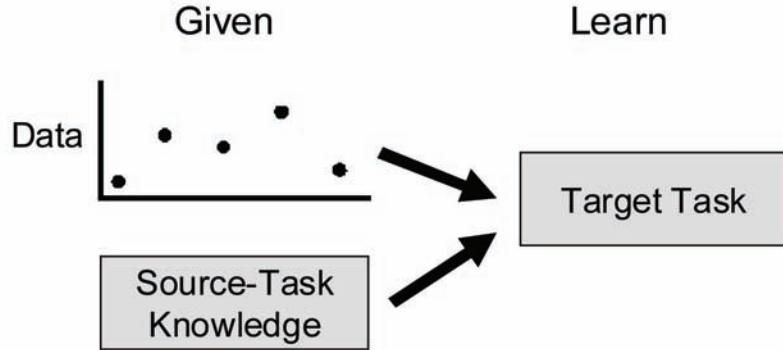
Human learners appear to have inherent ways to transfer knowledge between tasks. That is, we recognize and apply relevant knowledge from previous learning experiences when we encounter new tasks. The more related a new task is to our previous experience, the more easily we can master it.

Common machine learning algorithms, in contrast, traditionally address isolated tasks. Transfer learning attempts to improve on traditional machine learning by transferring knowledge learned in one or more source tasks and using it to improve learning in a related target task (see Figure 1). Techniques that enable knowledge transfer represent progress towards making machine learning as efficient as human learning.

This chapter provides an introduction to the goals, settings, and challenges of transfer learning. It surveys current research in this area, giving an overview of the state of the art and outlining the open problems.

DOI: 10.4018/978-1-60566-766-9.ch011

Figure 1. Transfer learning is machine learning with an additional source of information apart from the standard training data: knowledge from one or more related tasks.



Transfer methods tend to be highly dependent on the machine learning algorithms being used to learn the tasks, and can often simply be considered extensions of those algorithms. Some work in transfer learning is in the context of inductive learning, and involves extending well-known classification and inference algorithms such as neural networks, Bayesian networks, and Markov Logic Networks. Another major area is in the context of reinforcement learning, and involves extending algorithms such as Q-learning and policy search. This chapter surveys these areas separately.

The goal of transfer learning is to improve learning in the target task by leveraging knowledge from the source task. There are three common measures by which transfer might improve learning. First is the initial performance achievable in the target task using only the transferred knowledge, before any further learning is done, compared to the initial performance of an ignorant agent. Second is the amount of time it takes to fully learn the target task given the transferred knowledge compared to the amount of time to learn it from scratch. Third is the final performance level achievable in the target task compared to the final level without transfer. Figure 2 illustrates these three measures.

Figure 2. Three ways in which transfer might improve learning: a higher performance at the very beginning of learning, a steeper slope in the learning curve, or a higher asymptotic performance.

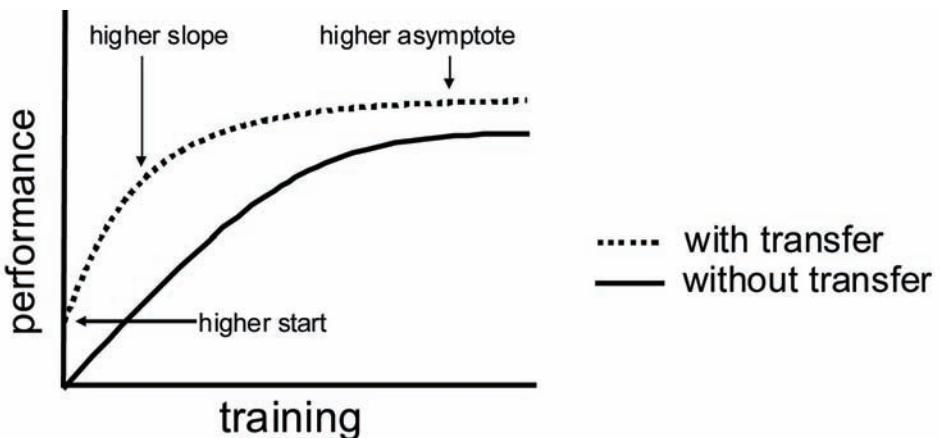
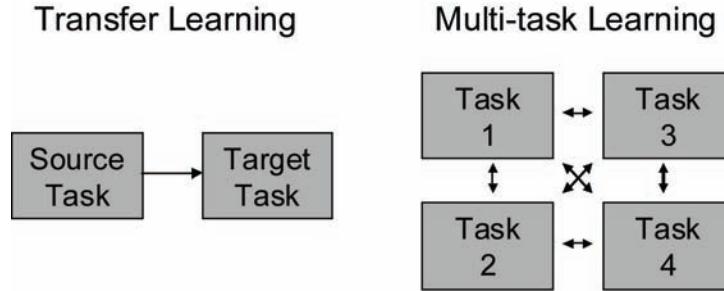


Figure 3. As we define transfer learning, the information flows in one direction only, from the source task to the target task. In multi-task learning, information can flow freely among all tasks.



If a transfer method actually decreases performance, then negative transfer has occurred. One of the major challenges in developing transfer methods is to produce positive transfer between appropriately related tasks while avoiding negative transfer between tasks that are less related. A section of this chapter discusses approaches for avoiding negative transfer.

When an agent applies knowledge from one task in another, it is often necessary to map the characteristics of one task onto those of the other to specify correspondences. In much of the work on transfer learning, a human provides this mapping, but there are some methods for performing the mapping automatically. A section of this chapter discusses automatic task mapping.

We will make a distinction between transfer learning and multi-task learning (Caruana, 1997), in which several tasks are learned simultaneously (see Figure 3). Multi-task learning is clearly closely related to transfer, but it does not involve designated source and target tasks; instead the learning agent receives information about several tasks at once. In contrast, by our definition of transfer learning, the agent knows nothing about a target task (or even that there will be a target task) when it learns a source task. It may be possible to approach a multi-task learning problem with a transfer-learning method, but the reverse is not possible. It is useful to make this distinction because a learning agent in a real-world setting is more likely to encounter transfer scenarios than multi-task scenarios.

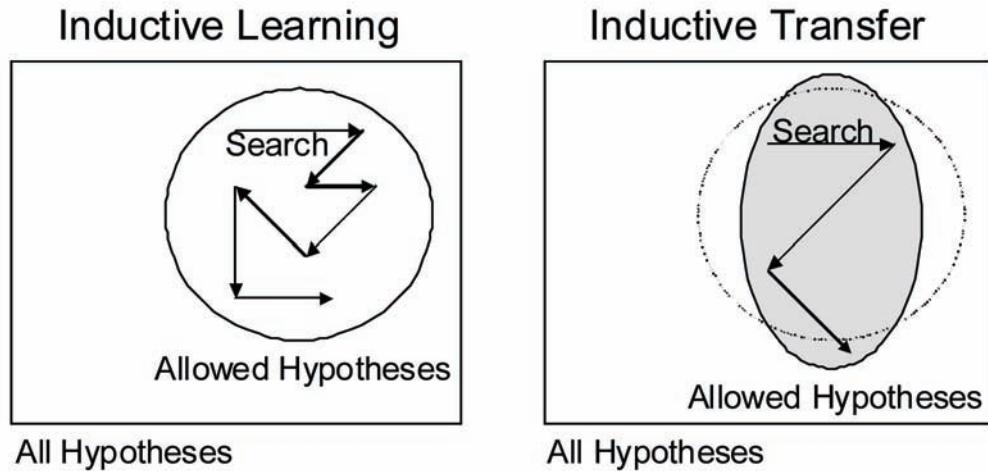
TRANSFER IN INDUCTIVE LEARNING

In an inductive learning task, the objective is to induce a predictive model from a set of training examples (Mitchell, 1997). Often the goal is classification, i.e. assigning class labels to examples. Examples of classification systems are artificial neural networks and symbolic rule-learners. Another type of inductive learning involves modeling probability distributions over interrelated variables, usually with graphical models. Examples of these systems are Bayesian networks and Markov Logic Networks (Richardson and Domingos, 2006).

The predictive model produced by an inductive learning algorithm should make accurate predictions not just on the training examples, but also on future examples that come from the same distribution. In order to produce a model with this generalization capability, a learning algorithm must have an inductive bias (Mitchell, 1997) – a set of assumptions about the concept being learned.

The bias of an algorithm is often based on the hypothesis space of possible models that it considers. For example, the hypothesis space of the naive Bayes model is limited by the assumption that example

Figure 4. Inductive learning can be viewed as a directed search through a specified hypothesis space (Mitchell, 1997). Inductive transfer uses source-task knowledge to adjust the inductive bias, which could involve changing the hypothesis space or the search steps.



characteristics are conditionally independent given the class of an example. The bias of an algorithm can also be based on its search process through the hypothesis space, which determines the order in which hypotheses are considered. For example, rule-learning algorithms typically construct rules one predicate at a time, which reflects the assumption that predicates contribute significantly to example coverage by themselves rather than in pairs or larger sets. These assumptions may not be correct, but they allow the inductive learner to generalize and make predictions on new data.

Transfer in inductive learning works by allowing source-task knowledge to affect the target task's inductive bias. It is concerned either with improving the speed of learning or with improving the generalization capability of the learned model. The next subsection discusses inductive transfer, and the following ones elaborate on three specific settings that have been popular in research on inductive transfer.

There is some related work that is not discussed here because it specifically addresses multi-task learning. For example, Niculescu-Mizil and Caruana (2007) learn Bayesian networks simultaneously for multiple related tasks by biasing learning toward similar structures for each task. While this is clearly related to transfer learning, it is not directly applicable to the scenario in which a target task is encountered after one or more source tasks have already been learned.

Inductive Transfer

In inductive transfer methods, the target-task inductive bias is chosen or adjusted based on source-task knowledge (see Figure 4). The way this is done varies depending on which inductive learning algorithm is used to learn the tasks. Some transfer methods narrow the hypothesis space, limiting the possible models, or remove search steps from consideration. Other methods broaden the hypothesis space, allowing the search to discover more complex models, or add new search steps.

Baxter (2000) frames the transfer problem as that of choosing one hypothesis space from a family of spaces. By solving a set of related source tasks in each hypothesis space of the family and determin-

ing which one produces the best overall generalization error, he selects the most promising space in the family for a target task. Baxter's work, unlike most in transfer learning, includes theoretical as well as experimental results. He derives bounds on the number of source tasks and examples needed to learn an inductive bias, and on the generalization capability of a target-task solution given the number of source tasks and examples in each task.

Thrun and Mitchell (1995) look at solving Boolean classification tasks in a lifelong-learning framework, where an agent encounters a collection of related problems over its lifetime. They learn each new task with a neural network, but they enhance the standard gradient-descent algorithm with slope information acquired from previous tasks. This speeds up the search for network parameters in a target task and biases it towards the parameters for previous tasks.

Mihalkova and Mooney (2006) perform transfer between Markov Logic Networks. Given a learned MLN for a source task, they learn an MLN for a related target task by starting with the source-task one and diagnosing each formula, adjusting ones that are too general or too specific in the target domain. The hypothesis space for the target task is therefore defined in relation to the source-task MLN by the operators that generalize or specify formulas.

Hlynsson (2007) phrases transfer learning in classification as a minimum description length problem given source-task hypotheses and target-task data. That is, the chosen hypothesis for a new task can use hypotheses for old tasks but stipulate exceptions for some data points in the new task. This method aims for a tradeoff between accuracy and compactness in the new hypothesis.

Ben-David and Schuller (2003) propose a transformation framework to determine how related two Boolean classification tasks are. They define two tasks as related with respect to a class of transformations if they are equivalent under that class; that is, if a series of transformations can make one task look exactly like the other.

Bayesian Transfer

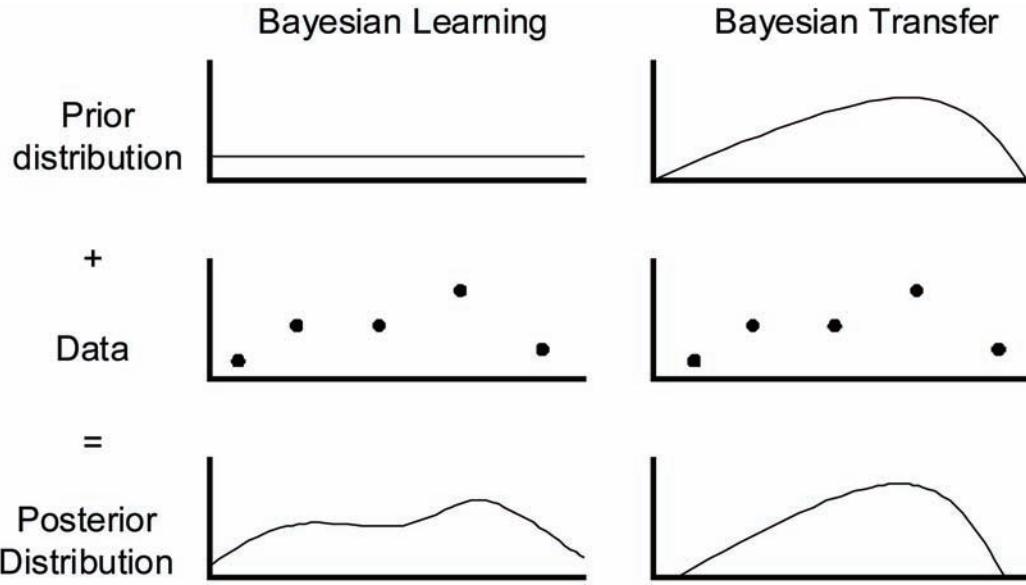
Bayesian learning methods are a popular setting for inductive transfer. Bayesian learning involves modeling probability distributions of variables, and simplifying models by taking advantage of conditional independence among variables. Bayesian models also make use of prior distributions, which describe the initial assumptions one can make about the distribution of variables in a domain before seeing any training data. Given the training data, a Bayesian model adjusts the prior distribution to produce a posterior distribution. It can then make predictions about future data using the posterior distribution. A strong prior can significantly affect a Bayesian model (see Figure 5). This provides a natural way for Bayesian learning methods to incorporate prior knowledge – in the case of transfer learning, source-task knowledge.

Marx et al. (2005) use a Bayesian transfer method for tasks solved by a logistic regression classifier. The usual prior for this classifier is a Gaussian distribution with a mean and variance set through cross-validation. To perform transfer, they instead estimate the mean and variance by averaging over several source tasks.

Raina et al. (2006) use a similar approach for multi-class classification by computing a multivariate Gaussian prior from several source tasks.

Dai et al. (2007a) apply a Bayesian transfer method to a Naive Bayes classifier. They set the initial probability parameters based on a single source task, and revise them using target-task data. They also provide some theoretical bounds on the prediction error and convergence rate of their algorithm.

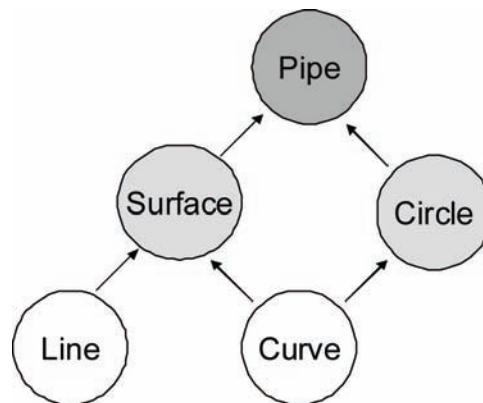
Figure 5. Bayesian learning uses a prior distribution to smooth the estimates from training data. Bayesian transfer can allow source-task knowledge to provide a more informative prior.



Hierarchical Transfer

Another popular setting for transfer in inductive learning is hierarchical transfer. In this setting, solutions to simple tasks are combined or provided as tools to produce a solution to a more complex task (see Figure 6). This can involve many tasks of varying complexity, rather than just a single source and target. The target task might use entire source-task solutions as parts of its own, or it might use them in a more subtle way to improve learning.

Figure 6. An example of a concept hierarchy that could be used for hierarchical transfer. Here the simple tasks involve recognizing lines and curves in images, and the more complex tasks use these to recognize surfaces, circles, and finally pipe shapes.



Sutton and McCallum (2005) begin with a sequential approach where the prediction for each task is used as a feature when learning the next task. They then proceed to turn the problem into a multi-task learning problem by combining all the models and applying them jointly, which brings their method outside our definition of transfer learning, but the initial sequential approach is an example of hierarchical transfer.

Stracuzzi (2006) looks at the problem of choosing relevant source-task Boolean concepts from a knowledge base to use while learning more complex concepts. He learns rules to express concepts from a stream of examples, allowing existing concepts to be used if they help to classify the examples, and adds and removes dependencies between concepts in the knowledge base.

Taylor et al. (2007) propose a transfer hierarchy that orders tasks by difficulty, so that an agent can learn them in sequence via inductive transfer. By putting tasks in order of increasing difficulty, they aim to make transfer more effective. This approach may be more applicable to the multi-task learning scenario, since by our definition of transfer learning the agent may not be able to choose the order in which it learns tasks, but it could be applied to help choose from an existing set of source tasks.

Transfer with Missing Data or Class Labels

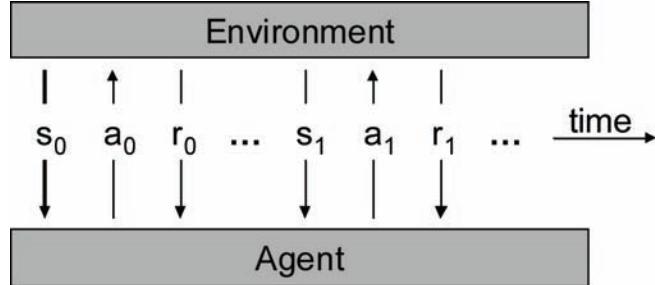
Inductive transfer can be viewed not only as a way to improve learning in a standard supervised-learning task, but also as a way to offset the difficulties posed by tasks that involve unsupervised learning, semi-supervised learning, or small datasets. That is, if there are small amounts of data or class labels for a task, treating it as a target task and performing inductive transfer from a related task can produce more accurate models. These approaches use source-task data to address limitations in the target-task data, even though the two datasets may come from somewhat different distributions.

The Bayesian transfer methods of Dai et al. (2007a) and Raina et al. (2006) are intended to compensate for small amounts of target-task data. One of the benefits of Bayesian learning is the stability that a prior distribution can provide in the absence of large datasets. By estimating a prior from related source tasks, these approaches prevent the overfitting that would tend to occur with limited target-task data.

Dai et al. (2007b) address transfer learning in a boosting algorithm, using large amounts of data from a previous task to supplement small amounts of new data. Boosting is a technique for learning several weak classifiers and combining them to form a stronger classifier (Freund and Schapire, 1997). After each classifier is learned, the examples are reweighted so that later classifiers focus more on examples the previous ones misclassified. Dai et al. extend this principle by also weighting source-task examples according to their similarity to target-task examples. This allows the algorithm to leverage source-task data that is applicable to the target task while paying less attention to data that appears less useful.

Shi et al. (2008) look at transfer learning in unsupervised and semi-supervised settings. They assume that a reasonably-sized dataset exists in the target task, but it is largely unlabeled due to the expense of having an expert assign labels. To address this problem they propose an active learning approach, in which the target-task learner requests labels for examples only when necessary. They construct a classifier with labeled examples, including mostly source-task ones, and estimate the confidence with which this classifier can label the unknown examples. When the confidence is too low, they request an expert label.

Figure 7. A reinforcement learning agent interacts with its environment: it receives information about its state (s), chooses an action to take (a), receives a reward (r), and then repeats.



TRANSFER IN REINFORCEMENT LEARNING

A reinforcement learning (RL) agent operates in a sequential-control environment called a Markov decision process (MDP) (Sutton and Barto, 1998). It senses the state of the environment and performs actions that change the state and also trigger rewards. Its objective is to learn a policy for acting in order to maximize its cumulative reward. This involves solving a temporal credit-assignment problem, since an entire sequence of actions may contribute to a single reward received at the end.

A typical RL agent behaves according to the diagram in Figure 7. At time step t , it observes the current state s_t and consults its current policy π in order to choose an action, $\pi(s_t) = a_t$. After taking the action, it receives a reward r_t and observes the new state s_{t+1} , and it uses that information to update its policy before repeating the cycle. Often RL consists of a sequence of episodes, which end whenever the agent reaches one of a set of ending states.

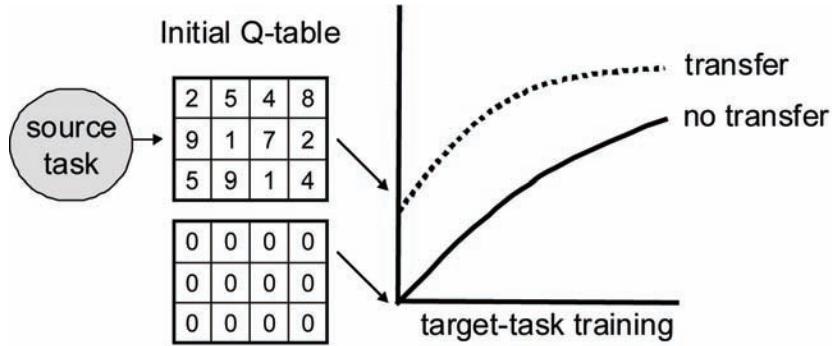
During learning, the agent must balance between exploiting the current policy (acting in areas that it knows to have high rewards) and exploring new areas to find potentially higher rewards. A common solution is the ϵ -greedy method, in which the agent takes random exploratory actions a small fraction of the time ($\epsilon \ll 1$), but usually takes the action recommended by the current policy.

There are several categories of RL algorithms. Some types of methods are only applicable when the agent knows its environment model (the reward function and the state transition function). In this case, dynamic programming can solve directly for the optimal policy without requiring any interaction with the environment. In most RL problems, however, the model is unknown. Model-learning approaches use interaction with the environment to build an approximation of the true model. Model-free approaches learn to act without ever explicitly modeling the environment.

Temporal-difference methods (Sutton, 1988) operate by maintaining and iteratively updating value functions to predict the rewards earned by actions. They begin with an inaccurate function and update it based on interaction with the environment, propagating reward information back along action sequences. One popular example is Q-learning (Watkins, 1989), which involves learning a function $Q(s, a)$ that estimates the cumulative reward starting in state s and taking action a and following the current policy thereafter. Given the optimal Q-function, the optimal policy is to take the action corresponding to $\text{argmax}_a Q(s, a)$. If there are small finite numbers of states and actions, the Q-function can be represented explicitly as a table. In domains that have large or infinite state spaces, a function approximator such as a neural network or support-vector machine can be used to represent the Q-function.

Policy-search methods, rather than maintaining a value function upon which a policy is based, in-

Figure 8. Starting-point methods for RL transfer set the initial solution based on the source task in the hope of starting at a higher performance level than the typical initial solution would. In this example, a Q-function table is initialized to a source-task table, and the target-task performance begins at a level that is only reached after some training when beginning with a typical all-zero table.



stead maintain and update a policy directly. They begin with a suboptimal policy and update it based on interaction with the environment. Heuristic search and optimization through gradient descent are among the approaches that can be used in policy search.

Transfer in RL is concerned with speeding up the learning process, since RL agents can spend many episodes doing random exploration before acquiring a reasonable policy or value function. We divide RL transfer into five categories that represent progressively larger changes to existing RL algorithms. The subsections below describe those categories and present examples from published research.

Starting-Point Methods

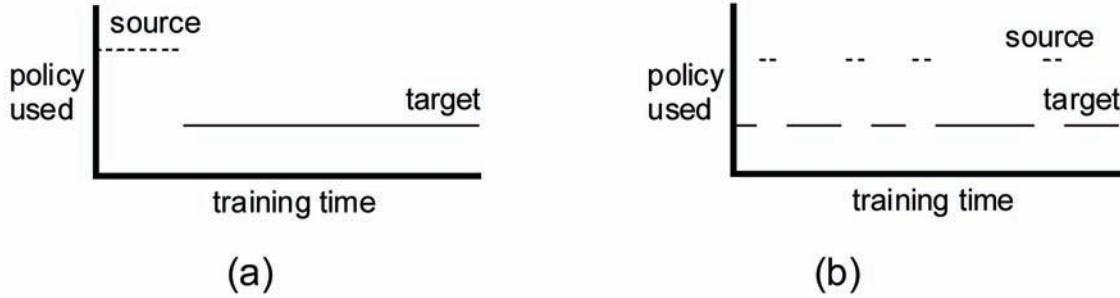
Since most RL methods begin with an initial solution and then update it through experience, one straightforward type of transfer in RL is to set the initial solution in a target task based on knowledge from a source task (see Figure 8). Compared to the random or zero setting that RL algorithms usually use at first, these starting-point methods can begin the RL process at a point much closer to a good target-task solution. There are variations on how to use the source-task knowledge to set the initial solution, but in general the RL algorithm in the target task is unchanged.

Taylor et al. (2005) use a starting-point method for transfer in temporal-difference RL. To perform transfer, they copy the final value function of the source task and use it as the initial one for the target task. This requires a mapping of features and actions between the tasks, which they provide manually based on their domain knowledge.

Tanaka and Yamamura (2003) use a similar approach in temporal-difference learning without function approximation, where value functions are simply represented by tables. This greater simplicity allows them to combine knowledge from several source tasks: they initialize the value table of the target task to the average of tables from several prior tasks. Furthermore, they use the standard deviations from prior tasks to determine priorities among temporal-difference backups.

Approaching temporal-difference RL as a batch problem instead of an incremental one allows for different kinds of starting-point transfer methods. In batch RL, the agent interacts with the environment for more than one step or episode at a time before updating its solution. Lazaric et al. (2008) perform

Figure 9. Imitation methods for RL transfer follow the source-task policy during some steps of the target task. The imitation steps may all occur at the beginning of the target task, as in (a), or they may be interspersed with steps that follow the developing target-task policy, as in (b).



transfer in this setting by finding source-task samples that are similar to the target task and adding them to the normal target-task samples in each batch, thus increasing the available data early on. The early solutions are almost entirely based on source-task knowledge, but the impact decreases in later batches as more target-task data becomes available.

Moving away from temporal-difference RL, starting-point methods can take even more forms. In a model-learning Bayesian RL algorithm, Wilson et al. (2007) perform transfer by treating the distribution of previous MDPs as a prior for the current MDP. In a policy-search genetic algorithm, Taylor et al. (2006) transfer a population of policies from a source task to serve as the initial population for a target task.

Imitation Methods

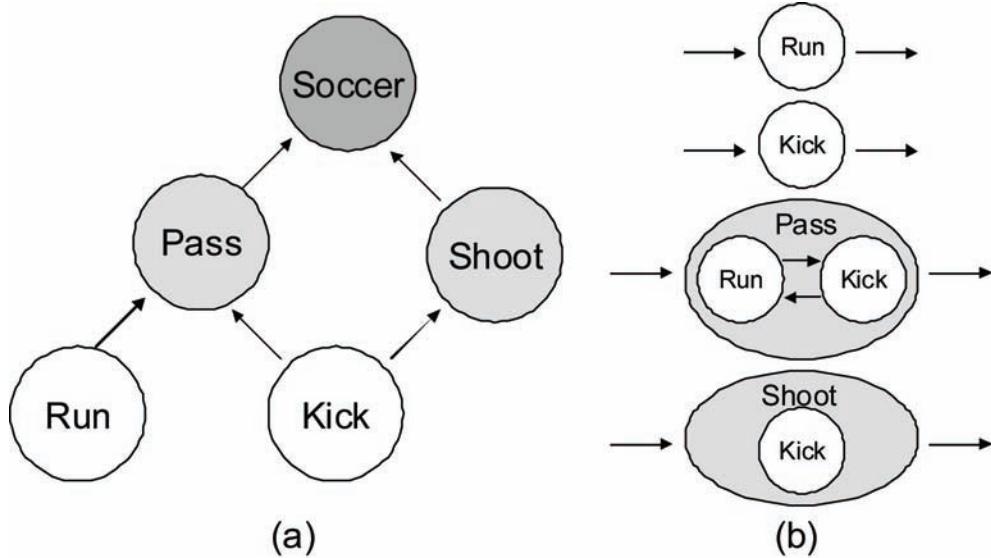
A second class of RL transfer methods involves applying the source-task policy to choose some actions while learning the target task. While they make no direct changes to the target-task solution the way that starting-point methods do, these imitation methods affect the developing solution by producing different function or policy updates. Compared to the random exploration that RL algorithms typically do, decisions based on a source-task policy can lead the agent more quickly to promising areas of the environment. There are variations in how the source-task policy is represented and in how heavily it is used in the target-task RL algorithm (see Figure 9).

One somewhat conservative method of this type is to follow a source-task policy only during exploration steps of the target task, when the agent would otherwise be taking a random action. Madden and Howley (2004) use this approach in tabular Q-learning. They represent a source-task policy as a set of rules in propositional logic and choose actions based on those rules during exploration steps.

Fernandez and Veloso (2006) give the RL agent a three-way choice between exploiting the current target-task policy, exploiting a past policy, and exploring randomly. They introduce a second parameter, in addition to the ϵ of ϵ -greedy exploration, to determine the probability of making each choice.

Another imitation method involves following a source-task policy for a fixed number of episodes at the beginning of the target task, providing a demonstration of potentially successful behavior, and then reverting to normal RL. In the early steps of the target task, the current policy can be so ill-formed that exploiting it is no different than exploring randomly. This approach aims to avoid that initial uncertainty and to generate enough data to create a reasonable target-task policy by the time the demonstration period

Figure 10. (a) An example of a task hierarchy that could be used to train agents to play soccer via hierarchical RL. Lower-level abilities like kicking a ball and running are needed for higher-level abilities like passing and shooting, which could then be combined to learn to play soccer. (b) The mid-level abilities represented as options alongside the low-level actions.



ends. Torrey et al. (2007) and (2008) perform transfer via demonstration, representing the source-task policy as a relational finite-state machine and a Markov Logic Network respectively.

Hierarchical Methods

A third class of RL transfer includes hierarchical methods. These view the source as a subtask of the target, and use the solution to the source as a building block for learning the target. Methods in this class have strong connections to the area of hierarchical RL, in which a complex task is learned in pieces through division into a hierarchy of subtasks (see Figure 10a).

One approach of this type is to compose several source-task solutions to form a target-task solution (Singh, 1992). Singh addresses a scenario in which complex tasks are temporal concatenations of simple ones, so that a target task can be solved by a composition of several smaller solutions.

Mehta et al. (2008) have a transfer method that works directly within the hierarchical RL framework. They learn a task hierarchy by observing successful behavior in a source task, and then use it to apply the MaxQ hierarchical RL algorithm (Dietterich, 2000) in the target task. This method uses transfer to removes the burden of designing a task hierarchy manually.

Other approaches operate within the framework of options, which is a term for temporally-extended actions in RL (Perkins and Precup, 1999). An option typically consists of a starting condition, an ending condition, and an internal policy for choosing lower-level actions. An RL agent treats each option as an additional action along with the original lower-level ones (see Figure 10b).

In some scenarios it may be useful to have the entire source-task policy as an option in the target task, as Croonenborghs et al. (2007) do. They learn a relational decision tree to represent the source-task

policy and allow the target-task learner to execute it as an option. Another possibility is to learn smaller options, either during or after the process of learning the source task, and make them available in the target. Asadi and Huber (2007) do this by finding frequently-visited states in the source task to serve as ending conditions for options.

Alteration Methods

A fourth class of RL transfer methods involves altering the state space, action space, or reward function of the target task based on source-task knowledge. These alteration methods have some overlap with option-based transfer, which also changes the action space in the target task, but they include other approaches as well.

One way to alter the target-task state space is to simplify it through state abstraction. Walsh et al. (2006) do this by aggregating over comparable source-task states. They use the aggregate states to learn the target task in a less complex environment.

There are also approaches that expand the target-task state space instead of reducing it. Taylor and Stone (2007a) do this by adding a new state variable in the target task. They learn a decision list that represents the source-task policy and use its output as the new state variable.

While option-based transfer methods add to the target-task action space, there is also some work in decreasing the action space. Sherstov and Stone (2005) do this by evaluating in the source task which of a large set of actions are most useful. They then consider only a smaller action set in the target task, which decreases the complexity of the value function significantly and also decreases the amount of exploration needed.

Reward shaping is a design technique in RL that aims to speed up learning by providing immediate rewards that are more indicative of cumulative rewards. Usually it requires human effort, as many aspects of RL task design do, but Konidaris and Barto (2006) do reward shaping automatically through transfer. They learn to predict rewards in the source task and use this information to create a shaped reward function in the target task.

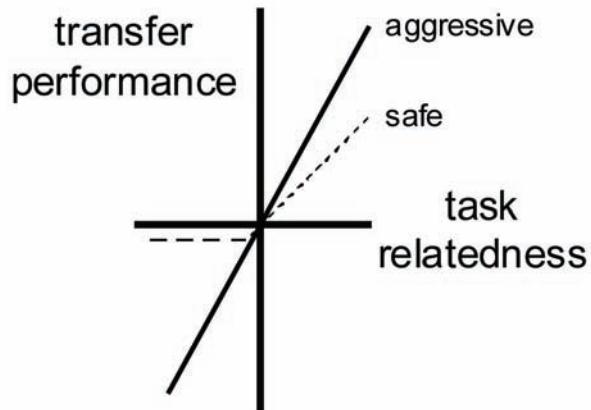
New RL Algorithms

A final class of RL transfer methods consists of entirely new RL algorithms. Rather than making small additions to an existing algorithm or making changes to the target task, these approaches address transfer as an inherent part of RL. They incorporate prior knowledge as an intrinsic part of the algorithm.

Price and Boutilier (1999) propose a temporal-difference algorithm in which value functions are influenced by observations of expert agents. They use a variant of the usual value-function update calculation that includes an expert's experience, weighted by the agent's confidence in itself and in the expert. They also perform extra backups at states the expert visits to focus attention on those areas of the state space.

There are several algorithms for case-based RL that accommodate transfer. Sharma et al. (2007) propose one in which Q-functions are estimated using a Gaussian kernel over stored cases in a library. Cases are added to the library from both the source and target tasks when their distance to their nearest neighbor is above a threshold. Taylor et al. (2008a) use source-task examples more selectively in their case-based RL algorithm. They use target-task cases to make decisions when there are enough, and only use source-task examples when insufficient target examples exist.

Figure 11. A representation of how the degree of relatedness between the source and target tasks translates to target-task performance when conducting transfer from the source task. With aggressive approaches, there can be higher benefits at high degrees of relatedness, but there can also be negative transfer at low levels. Safer approaches may limit negative transfer at the lower end, but may also have fewer benefits at the higher end.



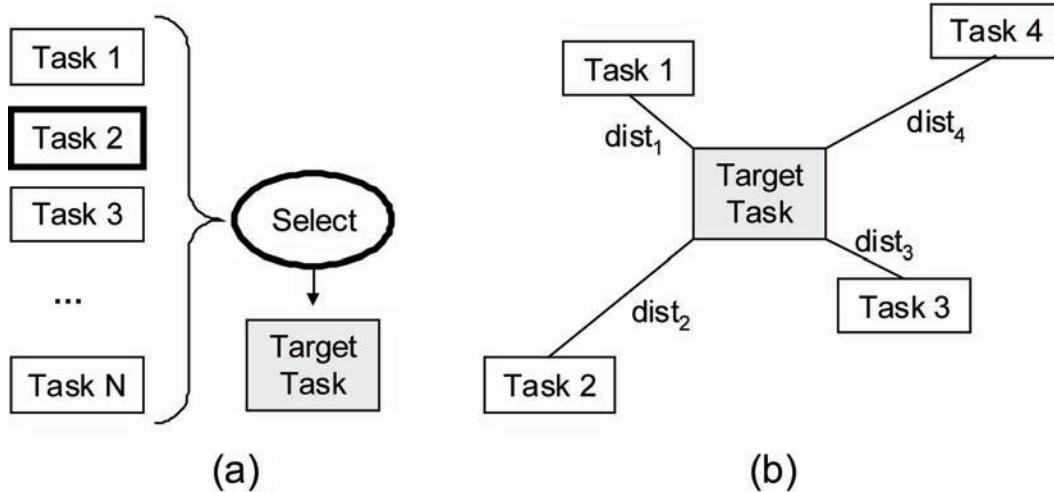
Torrey et al. (2005) and (2006) use an RL algorithm called Knowledge-Based Kernel Regression (KBKR) that allows transfer via advice-taking. Advice in this algorithm is a rule telling the agent which action to prefer in a specified set of states. KBKR uses a support-vector machine (SVM) to approximate the Q-function, and includes advice as a soft constraint in the SVM. The Q-function, which is relearned in batches using temporal-difference updates, trades off between matching the agent's experience and matching the advice. To perform transfer, advice is generated through automated analysis of the source-task solution; in (2005) it is constructed in propositional logic from the source-task Q-function, and in (2006) it is learned in first-order logic by observing the source-task agent as it follows its policy.

AVOIDING NEGATIVE TRANSFER

Given a target task, the effectiveness of any transfer method depends on the source task and how it is related to the target. If the relationship is strong and the transfer method can take advantage of it, the performance in the target task can significantly improve through transfer. However, if the source task is not sufficiently related or if the relationship is not well leveraged by the transfer method, the performance with many approaches may not only fail to improve – it may actually decrease, producing negative transfer. This section examines work on preventing transfer from negatively affecting performance.

Ideally, a transfer method would produce positive transfer between appropriately related tasks while avoiding negative transfer when the tasks are not a good match. In practice, these goals are difficult to achieve simultaneously. Approaches that have safeguards to avoid negative transfer often produce a smaller effect from positive transfer due to their caution. Conversely, approaches that transfer aggressively and produce large positive-transfer effects often have no protection against negative transfer (see Figure 11).

Figure 12. (a) One way to avoid negative transfer is to choose a good source task from which to transfer. In this example, Task 2 is selected as being the most related. (b) Another way to avoid negative transfer is to model the way source tasks are related to the target task and combine knowledge from them with those relationships in mind.



For example, consider the imitation methods for RL transfer. On one end of the range an agent imitates a source-task policy only during infrequent exploration steps, and on the other end it demonstrates the source-task policy for a fixed number of initial episodes. The exploration method is cautious and therefore unlikely to produce negative transfer, but it is also unlikely to produce large initial performance increases. The demonstration method is aggressive; if the source-task policy is a poor one for the target task, following it blindly will produce negative transfer. However, when the source-task solution is a decent one for the target task, it can produce some of the largest initial performance improvements of any method.

Rejecting Bad Information

One way of approaching negative transfer is to attempt to recognize and reject harmful source-task knowledge while learning the target task. The goal in this approach is to minimize the impact of bad information, so that the transfer performance is at least no worse than learning the target task without transfer. At the extreme end, an agent might disregard the transferred knowledge completely, but some methods also allow it to selectively reject parts and keep other parts.

Option-based transfer in reinforcement learning (Croonenborghs et al., 2007) is an example of an approach that naturally incorporates the ability to reject bad information. Since options are treated as additional actions, the agent can choose to use them or not to use them; in Q-learning, for example, agents learn Q-values for options just as for native actions. If an option regularly produces poor performance, its Q-values will degrade and the agent will choose it less frequently. However, if an option regularly leads to good results, its Q-values will grow and the agent will choose it more often. Option-based transfer can therefore provide a good balance between achieving positive transfer and avoiding negative transfer.

A specific approach that incorporates the ability to reject bad information is the KBKR advice-taking algorithm for transfer in reinforcement learning (Torrey et al., 2005). KBKR approximates the Q-function with a support-vector machine and includes advice from the source task as a soft constraint. Since the Q-function trades off between matching the agent's experience and matching the advice, the agent can learn to disregard advice that disagrees with its experience.

Rosenstein et al. (2005) present an approach for detecting negative transfer in naive Bayes classification tasks. They learn a hyperprior for both the source and target tasks, and the variance of this hyperprior is proportional to the dissimilarity between the tasks. It may be possible to use a method like this to decide whether to transfer at all, by setting an acceptable threshold of similarity.

Choosing a Source Task

There are more possibilities for avoiding negative transfer if there exists not just one source task, but a set of candidate source tasks. In this case the problem becomes choosing the best source task (see Figure 12a). Transfer methods without much protection against negative transfer may still be effective in this scenario, as long as the best source task is at least a decent match.

An example of this approach is the previously-mentioned transfer hierarchy of Taylor et al. (2007), who order tasks by difficulty. Appropriate source tasks are usually less difficult than the target task, but not so much simpler that they contain little information. Given a task ordering, it may be possible to locate the position of the target task in the hierarchy and select a source task that is only moderately less difficult.

Talvitie and Singh (2007) use a straightforward method of selecting a previous Markov decision process to transfer. They run each candidate MDP in the target task for a fixed length of time and order them by their performance. Then they select the best one and continue with it, only proceeding down the list if the current MDP begins performing significantly worse than it originally appeared. This trial-and-error approach, though it may be costly in the aggregate number of training episodes needed, is simple and widely applicable.

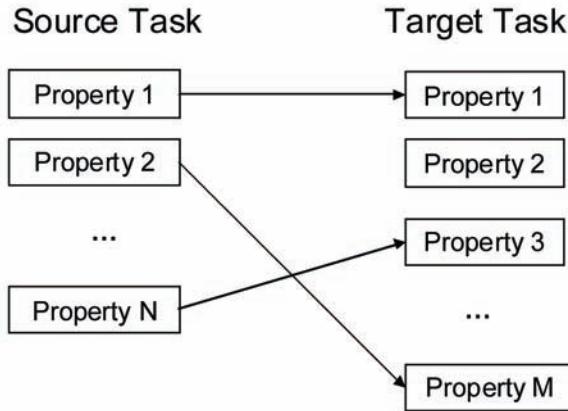
Kuhlmann and Stone (2007) look at finding similar tasks when each task is specified in a formal language. They construct a graph to represent the elements and rules of a task. This allows them to find identical tasks by checking for graph isomorphism, and by creating minor variants of a target-task graph, they can also search for similar tasks. If they find an isomorphic match, they conduct value-function transfer.

Eaton and DesJardins (2006) propose choosing from among candidate solutions to a source task rather than from among candidate source tasks. Their setting is multi-resolution learning, where a classification task is solved by an ensemble of models that vary in complexity. Low-resolution models are simple and coarse, while higher-resolution models are more complex and detailed. They reason that high-resolution models are less transferrable between tasks, and select a resolution below which to share models with a target task.

Modeling Task Similarity

Given multiple candidate source tasks, it may be beneficial to use several or all of them rather than to choose just one (see Figure 12b). Some approaches discussed in this chapter do this naively, without evaluating how the source tasks are related to the target. However, there are some approaches that ex-

Figure 13. A mapping generally translates source-task properties into target-task properties. The numbers of properties may not be equal in the two tasks, and the mapping may not be one-to-one. Properties may include entries in a feature vector, objects in a relational world, RL actions, etc.



plicitly model relationships between tasks and include this information in the transfer method. This can lead to better use of source-task knowledge and decrease the risk of negative transfer.

Carroll and Seppi (2005) develop several similarity measures for reinforcement learning tasks, comparing policies, value functions, and rewards. These are only measurable while the target task is being learned, so their practical use in transfer scenarios is limited. However, they make the relevant point that task similarity is intimately linked with a particular transfer method, and cannot be evaluated independently.

Eaton et al. (2008) construct a graph in which nodes represent source tasks and distances represent a transferability metric. Given a new inductive learning task, they estimate parameters by fitting the task into the graph and learning a function that translates graph locations to task parameters. This method not only models the relationships between tasks explicitly, but also gives an algorithm for the informed use of several source tasks in transfer learning.

Ruckert and Kramer (2008) look at inductive transfer via kernel methods. They learn a meta-kernel that serves as a similarity function between tasks. Given this and a set of kernels that perform well in source tasks, they perform numerical optimization to construct a kernel for a target task. This approach determines the inductive bias (the kernel) in the target task by combining information from several source tasks whose relationships to the target are known.

AUTOMATICALLY MAPPING TASKS

An inherent aspect of transfer learning is recognizing the correspondences between tasks. Knowledge from one task can only be applied to another if it is expressed in a way that the target-task agent understands. In some cases, the representations of the tasks are assumed to be identical, or at least one is a subset of the other. Otherwise, a mapping is needed to translate between task representations (see Figure 13).

Many transfer approaches do not address the mapping problem directly and require that a human provide this information. However, there are some transfer approaches that do address the mapping problem. This section discusses some of this work.

Equalizing Task Representations

For some transfer scenarios, it may be possible to avoid the mapping problem altogether by ensuring that the source and target tasks have the same representation. If the language of the source-task knowledge is identical to or a subset of the language of the target task, it can be applied directly with no translation. Sometimes a domain can be constructed so that this occurs naturally, or a common representation that equalizes the tasks can be found.

Relational learning is useful for creating domains that naturally produce common task representations. First-order logic represents objects in a domain with symbolic variables, which can allow abstraction that the more typical propositional feature vector cannot. Driessens et al. (2006) show how relational reinforcement learning can simplify transfer in RL.

Another framework for constructing a domain relationally is that of Konidaris and Barto (2006), who express knowledge in two different spaces. In agent space the representation is constant across tasks, while in problem space it is task-dependent. They transfer agent-space knowledge only because its common representation makes it straightforward to transfer.

Pan et al. (2008) take a mathematical approach to finding a common representation for two separate classification tasks. They use kernel methods to find a low-dimensional feature space where the distributions of source and target data are similar, and transfer a source-task model for this smaller space. This approach stretches our strict definition of transfer learning, which assumes the target task is unknown when the source task is learned, but in some scenarios it may be practical to adjust the source-task solution to a different feature space after gaining some knowledge about the target task.

Trying Multiple Mappings

One straightforward way of solving the mapping problem is to generate several possible mappings and allow the target-task agent to try them all. The candidate mappings can be an exhaustive set, or they can be limited by constraints on what elements are permissible matches for other elements. Exhaustive sets may be computationally infeasible for large domains.

Taylor et al. (2008b) perform an exhaustive search of possible mappings in RL transfer. They evaluate each candidate using a small number of episodes in the target task, and select the best one to continue learning. Mihalkova et al. (2007) limit their search for mappings in MLN transfer, requiring that mapped predicates have matching arity and argument types. Under those constraints, they conduct an exhaustive search to find the best mapping between networks.

Soni and Singh (2006) not only limit the candidate mappings by considering object types, but also avoid a separate evaluation of each mapping by using options in RL transfer. They generate a set of possible mappings by connecting target-task objects to all source-task objects of the matching type. With each mapping, they create an option from a source MDP. The options framework gives an inherent way to compare multiple mappings while learning a target MDP without requiring extra trial periods.

Mapping by Analogy

If the task representations must differ, and the scenario calls for choosing one mapping rather than trying multiple candidates, then there are some methods that construct a mapping by logical analogy. These methods examine the characteristics of the source and target tasks and find elements that correspond. For example, in reinforcement learning, actions that correspond produce similar rewards and state changes, and objects that correspond are affected similarly by actions.

Analogical structure mapping (Falkenhainer et al., 1989) is a generic procedure based on cognitive theories of analogy that finds corresponding elements. It assigns scores to local matches and searches for a global match that maximizes the scores; permissible matches and scoring functions are domain-dependent. Several transfer approaches use this framework solve the mapping problem. Klenk and Forbus (2007) apply it to solve physics problems that are written in a predicate-calculus language by retrieving and forming analogies from worked solutions written in the same language. Liu and Stone (2006) apply it in reinforcement learning to find matching features and actions between tasks.

There are also some approaches that rely more on statistical analysis than on logical reasoning to find matching elements. Taylor and Stone (2007b) learn mappings for RL tasks by running a small number of target-task episodes and then training classifiers to characterize actions and objects. If a classifier trained for one action predicts the results of another action well, then those actions are mapped; likewise, if a classifier trained for one object predicts the behavior of another object well, those objects are mapped. Wang and Mahadevan (2008) translate datasets to low-dimensional feature spaces using dimensionality reduction, and then perform a statistical shaping technique called Procrustes analysis to align the feature spaces.

THE FUTURE OF TRANSFER LEARNING

The challenges discussed in this chapter will remain relevant in future work on transfer learning, particularly the avoidance of negative transfer and the automation of task mapping. Humans appear to have natural mechanisms for deciding when to transfer information, selecting appropriate sources of knowledge, and determining the appropriate level of abstraction. It is not always clear how to make these decisions for a single machine learning algorithm, much less in general.

Another challenge for future work is to enable transfer between more diverse tasks. Davis and Domingos (2008) provide a potential direction for this in their work on MLN transfer. They perform pattern discovery in the source task to find second-order formulas, which represent universal abstract concepts like symmetry and transitivity. When learning an MLN for the target task, they allow the search to use the discovered formulas in addition to the original predicates in the domain. This approach is recognizable as inductive transfer, but the source-task knowledge is highly abstract, which allows the source and target tasks to differ significantly.

Yet another challenge is to perform transfer in more complex testbeds. Particularly in RL transfer, it can become much more difficult to achieve transfer as the source and target tasks become more complex. Since practical applications of reinforcement learning are likely to be highly complex, it is important not to limit research on RL tranfer to simple domains.

Transfer learning has become a sizeable subfield in machine learning. It has ideological benefits, because it is seen as an important aspect of human learning, and also practical benefits, because it can

make machine learning more efficient. As computing power increases and researchers apply machine learning to more and more complex problems, abilities like knowledge transfer can only become more desirable.

ACKNOWLEDGMENT

This chapter was written while the authors were partially supported by DARPA grants HR0011-07-C-0060 and FA8650-06-C-7606.

REFERENCES

- Asadi, M., & Huber, M. (2007). Effective control knowledge transfer through learning skill and representation hierarchies. In *Proceedings of the International Joint Conference on Artificial Intelligence*.
- Baxter, J. (2000). A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12, 149–198.
- Ben-David, S., & Schuller, R. (2003). Exploiting task relatedness for multiple task learning. In *Proceedings of the Conference on Learning Theory*.
- Carroll, C., & Seppi, K. (2005). Task similarity measures for transfer in reinforcement learning task libraries. In *Proceedings of the IEEE International Joint Conference on Neural Networks*.
- Caruana, R. (1997). Multitask learning. *Machine Learning*, 28, 41–75. doi:10.1023/A:1007379606734
- Croonenborghs, T., Driessens, K., & Bruynooghe, M. (2007). Learning relational skills for inductive transfer in relational reinforcement learning. In *Proceedings of the International Conference on Inductive Logic Programming*.
- Dai, W., Xue, G., Yang, Q., & Yu, Y. (2007a). Transferring Naive Bayes classifiers for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Dai, W., Yang, Q., Xue, G., & Yu, Y. (2007b). Boosting for transfer learning. In *Proceedings of the International Conference on Machine Learning*.
- Davis, J., & Domingos, P. (2008). Deep transfer via second-order Markov logic. In *Proceedings of the AAAI Workshop on Transfer Learning for Complex Tasks*.
- Dietterich, T. (2000). Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13, 227–303.
- Driessens, K., Ramon, J., & Croonenborghs, T. (2006). Transfer learning for reinforcement learning through goal and policy parametrization. In *Proceedings of the ICML Workshop on Structural Knowledge Transfer for Machine Learning*.
- Eaton, E., & DesJardins, M. (2006). Knowledge transfer with a multiresolution ensemble of classifiers. In *Proceedings of the ICML Workshop on Structural Knowledge Transfer for Machine Learning*.

- Eaton, E., DesJardins, M., & Lane, T. (2008). Modeling transfer relationships between learning tasks for improved inductive transfer. In *Proceedings of the European Conference on Machine Learning*.
- Falkenhainer, B., Forbus, K., & Gentner, D. (1989). The structure-mapping engine: Algorithm and examples. *Artificial Intelligence*, 41, 1–63. doi:10.1016/0004-3702(89)90077-5
- Fernandez, F., & Veloso, M. (2006). Probabilistic policy reuse in a reinforcement learning agent. In *Proceedings of the Conference on Autonomous Agents and Multi-Agent Systems*.
- Freund, Y., & Schapire, R. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1), 119–139. doi:10.1006/jcss.1997.1504
- Hlynsson, H. (2007). *Transfer learning using the minimum description length principle with a decision tree application*. Unpublished master's thesis, University of Amsterdam.
- Klenk, M., & Forbus, K. (2007). Measuring the level of transfer learning by an AP physics problem-solver. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Konidaris, G., & Barto, A. (2006). Autonomous shaping: Knowledge transfer in reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.
- Kuhlmann, G., & Stone, P. (2007). Graph-based domain mapping for transfer learning in general games. In *Proceedings of the European Conference on Machine Learning*.
- Lazaric, A., Restelli, M., & Bonarini, A. (2008). Transfer of samples in batch reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.
- Liu, Y., & Stone, P. (2006). Value-function-based transfer for reinforcement learning using structure mapping. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Madden, M., & Howley, T. (2004). Transfer of experience between reinforcement learning environments with progressive difficulty. *Artificial Intelligence Review*, 21, 375–398. doi:10.1023/B:AIRE.0000036264.95672.64
- Marx, Z., Rosenstein, M., Kaelbling, L., & Dietterich, T. (2005). Transfer learning with an ensemble of background tasks. In *Proceedings of the NIPS Workshop on Transfer Learning*.
- Mehta, N., Ray, S., Tadepalli, P., & Dietterich, T. (2008). Automatic discovery and transfer of MAXQ hierarchies. In *Proceedings of the International Conference on Machine Learning*.
- Mihalkova, L., Huynh, T., & Mooney, R. (2007). Mapping and revising markov logic networks for transfer learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Mihalkova, L., & Mooney, R. (2006). Transfer learning with Markov logic networks. In *Proceedings of the ICML Workshop on Structural Knowledge Transfer for Machine Learning*.
- Mitchell, T. (1997). *Machine learning*. New York: McGraw-Hill.
- Niculescu-Mizil, A., & Caruana, R. (2007). Inductive transfer for Bayesian network structure learning. In *Proceedings of the Conference on AI and Statistics*.

- Pan, S., Kwok, J., & Yang, Q. (2008). Transfer learning via dimensionality reduction. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Perkins, T., & Precup, D. (1999). *Using options for knowledge transfer in reinforcement learning* (Tech. Rep. UM-CS-1999-034). University of Massachusetts, Amherst, USA.
- Price, B., & Boutilier, C. (1999). Implicit imitation in multiagent reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.
- Raina, R., Ng, A., & Koller, D. (2006). Constructing informative priors using transfer learning. In *Proceedings of the International Conference on Machine Learning*.
- Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine Learning*, 62(1-2), 107–136. doi:10.1007/s10994-006-5833-1
- Rosenstein, M., Marx, Z., Kaelbling, L., & Dietterich, T. (2005). To transfer or not to transfer. In *Proceedings of the NIPS Workshop on Inductive Transfer*.
- Ruckert, U., & Kramer, S. (2008). Kernel-based inductive transfer. In *Proceedings of the European Conference on Machine Learning*.
- Sharma, M., Holmes, M., Santamaria, J., Irani, A., Isbell, C., & Ram, A. (2007). Transfer learning in real-time strategy games using hybrid CBR/RL. In *Proceedings of the International Joint Conference on Artificial Intelligence*.
- Sherstov, A., & Stone, P. (2005). Action-space knowledge transfer in MDPs: Formalism, suboptimality bounds, and algorithms. In *Proceedings of the Conference on Learning Theory*.
- Shi, X., Fan, W., & Ren, J. (2008). Actively transfer domain knowledge. In *Proceedings of the European Conference on Machine Learning*.
- Singh, S. (1992). Transfer of learning by composing solutions of elemental sequential tasks. *Machine Learning*, 8(3-4), 323–339. doi:10.1007/BF00992700
- Soni, V., & Singh, S. (2006). Using homomorphisms to transfer options across continuous reinforcement learning domains. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Stracuzzi, D. (2006). Memory organization and knowledge transfer. In *Proceedings of the ICML Workshop on Structural Knowledge Transfer for Machine Learning*.
- Sutton, C., & McCallum, A. (2005). Composition of conditional random fields for transfer learning. In *Proceedings of the Conference on Empirical methods in Natural Language Processing*.
- Sutton, R. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, 3, 9–44.
- Sutton, R., & Barto, A. (1998). *Reinforcement learning: An introduction*. Cambridge, MA: MIT Press.
- Talvitie, E., & Singh, S. (2007). An experts algorithm for transfer learning. In *Proceedings of the International Joint Conference on Artificial Intelligence*.

- Tanaka, F., & Yamamura, M. (2003). Multitask reinforcement learning on the distribution of MDPs. *Transactions of the Institute of Electrical Engineers of Japan*, 123(5), 1004–1011.
- Taylor, M., Jong, N., & Stone, P. (2008a). Transferring instances for model-based reinforcement learning. In *Proceedings of the European Conference on Machine Learning*.
- Taylor, M., Kuhlmann, G., & Stone, P. (2007). Accelerating search with transferred heuristics. In *Proceedings of the ICAPS Workshop on AI Planning and Learning*.
- Taylor, M., Kuhlmann, G., & Stone, P. (2008b). Autonomous transfer for reinforcement learning. In *Proceedings of the Conference on Autonomous Agents and Multi-Agent Systems*.
- Taylor, M., & Stone, P. (2007a). Cross-domain transfer for reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.
- Taylor, M., & Stone, P. (2007b). Transfer via inter-task mappings in policy search reinforcement learning. In *Proceedings of the Conference on Autonomous Agents and Multi-Agent Systems*.
- Taylor, M., Stone, P., & Liu, Y. (2005). Value functions for RL-based behavior transfer: A comparative study. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Taylor, M., Whiteson, S., & Stone, P. (2006). Transfer learning for policy search methods. In *Proceedings of the ICML Workshop on Structural Knowledge Transfer for Machine Learning*.
- Thrun, S., & Mitchell, T. (1995). Learning one more thing. In *Proceedings of the International Joint Conference on Artificial Intelligence*.
- Torrey, L., Shavlik, J., Natarajan, S., Kuppili, P., & Walker, T. (2008). Transfer in reinforcement learning via Markov logic networks. In *Proceedings of the AAAI Workshop on Transfer Learning for Complex Tasks*.
- Torrey, L., Shavlik, J., Walker, T., & Maclin, R. (2006). Relational skill transfer via advice taking. In *Proceedings of the ICML Workshop on Structural Knowledge Transfer for Machine Learning*.
- Torrey, L., Shavlik, J., Walker, T., & Maclin, R. (2007). Relational macros for transfer in reinforcement learning. In *Proceedings of the International Conference on Inductive Logic Programming*.
- Torrey, L., Walker, T., Shavlik, J., & Maclin, R. (2005). Using advice to transfer knowledge acquired in one reinforcement learning task to another. In *Proceedings of the European Conference on Machine Learning*.
- Walsh, T., Li, L., & Littman, M. (2006). Transferring state abstractions between MDPs. In *Proceedings of the ICML Workshop on Structural Knowledge Transfer for Machine Learning*.
- Wang, C., & Mahadevan, S. (2008). Manifold alignment using Procrustes analysis. In *Proceedings of the International Conference on Machine Learning*.
- Watkins, C. (1989). *Learning from delayed rewards*. Unpublished doctoral dissertation, University of Cambridge.

Wilson, A., Fern, A., Ray, S., & Tadepalli, P. (2007). Multi-task reinforcement learning: A hierarchical Bayesian approach. In *Proceedings of the International Conference on Machine Learning*.

KEY TERMS AND DEFINITIONS

Transfer Learning: methods in machine learning that improve learning in a target task by transferring knowledge from one or more related source tasks.

Multi-Task Learning: methods in machine learning for learning multiple tasks simultaneously.

Source Task: a task from which knowledge is transferred to a target task.

Target Task: a task in which learning is improved through knowledge transfer from a source task.

Negative Transfer: a decrease in learning performance in a target task due to transfer learning.

Mapping: a description of the correspondences between properties of source and target tasks in transfer learning or analogical reasoning.

Inductive Learning: a type of machine learning in which a predictive model is induced from a set of training examples.

Inductive Bias: the set of assumptions that an inductive learner makes about the concept being learned.

Reinforcement Learning: a type of machine learning in which an agent learns, through its own experience, to navigate through an environment, choosing actions in order to maximize the sum of rewards.

Policy: the mechanism by which a reinforcement-learning agent chooses which action to execute next.

Chapter 12

Machine Learning in Personalized Anemia Treatment

Adam E. Gaweda
University of Louisville, USA

ABSTRACT

This chapter presents application of reinforcement learning to drug dosing personalization in treatment of chronic conditions. Reinforcement learning is a machine learning paradigm that mimics the trial-and-error skill acquisition typical for humans and animals. In treatment of chronic illnesses, finding the optimal dose amount for an individual is also a process that is usually based on trial-and-error. In this chapter, the author focuses on the challenge of personalized anemia treatment with recombinant human erythropoietin. The author demonstrates the application of a standard reinforcement learning method, called Q-learning, to guide the physician in selecting the optimal erythropoietin dose. The author further addresses the issue of random exploration in Q-learning from the drug dosing perspective and proposes a “smart” exploration method. Finally, the author performs computer simulations to compare the outcomes from reinforcement learning-based anemia treatment to those achieved by a standard dosing protocol used at a dialysis unit.

INTRODUCTION

Pharmacological treatment of chronic illnesses often resembles a trial-and-error process. A physician usually initiates the treatment with a standard dose and observes the patient for a response and / or dangerous side effects. If the response is not sufficient, the dose is increased. On the other hand, if a side effect occurs, the dose is decreased. A standardized dosing strategy for a specific drug is usually established during Phase II of a clinical trial. However, such a standardized strategy is usually based on population dose-response characteristics and may not lead to optimal outcomes on individual basis.

DOI: 10.4018/978-1-60566-766-9.ch012

Optimal outcomes are more likely to be achieved if the dosing is adjusted over time based on the individual response. This is especially the case for chronic illnesses, such as anemia of End Stage Renal Disease (ESRD).

Anemia of ESRD is one of the common conditions in patients receiving hemodialysis (Eschbach and Adamson, 1985). It is caused by insufficient red blood cell production due to unavailability of a hormone called erythropoietin, produced primarily by the kidneys. Untreated, anemia leads to a number of conditions including cardiovascular disease (Harnett et al., 1995), as well as decreased quality of life (Wolcott et al., 1989) and increased mortality (Lowrie et al., 1994). External administration of recombinant human erythropoietin (rHuEPO) is a standard form of treatment for the anemia of ESRD. In the United States, the anemia treatment is governed by the National Kidney Foundation guidelines which recommend that the Hemoglobin (Hgb), a surrogate marker of red blood cell level, in patients receiving rHuEPO be maintained between 11 and 12 g/dL. Based on these recommendations, dialysis units develop their own sets of dose adjustment rules governing rHuEPO dosages in relation to Hgb levels. However, these rules are again often derived from population outcome measures and do not always provide the optimal outcome. For example, it is known that at a given point in time 2/3 of the ESRD patients in the U.S. are outside of the NKF recommended target range (Lacson et al., 2003).

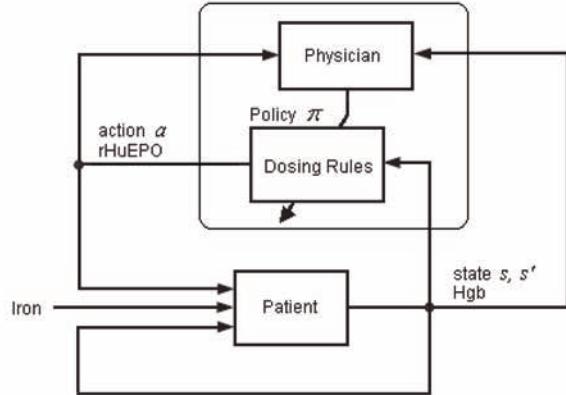
To facilitate the personalization of drug administration, we have proposed the application of a Machine Learning paradigm called Reinforcement Learning to anemia treatment (Gaweda et al., 2005). Reinforcement Learning mimics the goal-oriented skill acquisition performed by humans (Dayan and Balleine, 2002). It represents an intelligent system, referred to as an “agent,” which discovers what actions to take through interaction with its environment, by trying them and receiving rewards, in order to drive the environment to a “goal” state. Agent’s choices may affect the environment not only immediately, but also over the long term. The first reported attempt at using Reinforcement Learning for drug dosing has been described by Sinzinger and Moore (2005). They show that a Reinforcement Learning based control of patient sedation performs better than conventional closed-loop controllers, such as PID.

In (Gaweda et al., 2005), we applied a Reinforcement Learning method called *Q*-learning, to anemia treatment in an on-line fashion. We concluded that in its simplest form this method’s performance is comparable to that of a rHuEPO dosing protocol used at our dialysis unit. However, a pure trial-and-error based learning was found insufficient to warrant any further improvement in the primary outcome, defined as maintaining the Hgb within the 11-12 g/dL range. To alleviate this drawback, we focused our attention on adding an element of heuristic supervision to guide the *Q*-learning (Gaweda et al., 2007). Others (Martin-Guerrero et al., 2006) have also applied the concept Reinforcement Learning to anemia treatment in an off-line fashion, using *Q*-learning to elicit optimal dosing strategy for rHuEPO from historical data records.

One of the key issues of Reinforcement Learning is the trade-off between exploration and exploitation (Sutton and Barto, 1998). To maximize the reward, the agent should take actions that are known to result in large rewards (exploitation). However, to discover such actions, the agent must try actions that it has not selected before (exploration). Our efforts so far, as well as those by Martin-Guerrero et al. (2006) focused on a purely exploiting form of Reinforcement Learning. An obvious drawback of the exploiting learning is the likelihood of achieving suboptimal results.

In this chapter, building upon the work presented in (Gaweda et al., 2007), we describe a Reinforcement Learning approach to anemia treatment which combines the guided *Q*-learning with the exploratory component. We first describe the issue of random exploration and its potential pitfalls in the case of drug dosing. To eliminate the risk involved in exploring undesired dose amounts, we propose the concept of

Figure 1. Block diagram of a simple Reinforcement Learning system applied to anemia treatment.



“smart” exploration. To illustrate the operation of the algorithm and evaluate its performance, we perform numerical simulations of anemia treatment using pharmacodynamic model of an ESRD patient.

Reinforcement Learning and Drug Dosing

As mentioned in the Introduction, the agent and its environment are the two key components of a Reinforcement Learning system. The other essential elements are a policy, π , a reward function, r , and a value function $V(s)$ or $Q(s,a)$. A policy specifies agent’s action a (from a set of actions A), at a given state s (from a set of states S), of the environment. A reward function defines the goal behind agent’s interaction with the environment. In other words, it represents the state of the environment with a numerical value corresponding to the immediate desirability of the state. On the other hand, a value function describes the long term desirability of the state. Simply speaking, the value of a state $V(s)$ can be viewed as a cumulative expected reward after state s has been encountered. We can also consider value of an action $Q(s,a)$, which is the cumulative expected reward after the agent takes the action a upon encountering state s .

Translating the Reinforcement Learning framework into the realm of drug dosing, we would represent the physician as the agent and the patient as the environment. The dose adjustment strategy would correspond to the policy and the patient’s response to drug would be the state. The goal would be to establish such a dosing policy, which would result in the most desirable response. In the case of anemia treatment, such a desirable response would be the Hgb within the range of 11 to 12 g/dL.

Figure 1 presents a block diagram of a simple Reinforcement Learning system as applied to rHuEPO dosing in anemia treatment. The Hgb level in a patient (Environment) depends primarily on two factors, the administered rHuEPO dose and the Iron stores available for red blood cell production, typically measured by Transferrin Saturation (TSat) level as a surrogate marker. The physician’s task is then to come up with a set of rules (Policy) for rHuEPO dose adjustment (Action) in order to drive the Hgb (State) to, and maintain it within the target range in the long run.

The major advantage of the Reinforcement Learning methods compared to Control Engineering methods or Dynamic Programming is that former require no knowledge of the environment. All learning

occurs through the aforementioned interaction between the agent and the environment.

The Reinforcement Learning algorithms can belong to one of two general families, Monte Carlo or Temporal Difference. The Monte Carlo (MC) approach is based on rewards averaged over finite episodes. It is therefore well suited for episodic tasks, or tasks that can be divided into finite episodes. Temporal Difference (TD) methods, on the other hand, are capable of incremental learning in an on-line fashion. This makes them well suited for application to continuous tasks, such as drug dosing. The Reinforcement Learning methods can be further classified as on- or off-policy. Roughly speaking, an on-policy method is one that evaluates a policy that is used to select actions. An off-policy method, on the other hand, is capable to assess a policy, based on actions selected based on other policies. Compared to on-policy, the off-policy methods are known to show faster convergence to optimal solutions, which makes them well suited for control tasks. Perhaps the most well known off-policy TD method is Q -learning (Watkins, 1989). For a continuous (non-episodic) task, it can be represented in an algorithmic form as follows.

```

Initialize state-action values  $Q(s_i, a_i)$ ,  $s_i \in S$ ,  $a_i \in A$ 
Initialize state  $s$ 
Repeat
    Derive policy  $\pi$  from  $Q(s_i, a_i)$ 
    Choose action  $a$  based on  $s$  using  $\pi$ 
    Perform action  $a$ , observe next state  $s'$  and reward  $r$ 
    Update state-action value:
        
$$Q(s, a) \leftarrow Q(s, a) + \alpha [ r + \gamma \max_{a'} Q(s', a') - Q(s, a) ]$$

    Update state:
        
$$s \leftarrow s'$$


```

The symbol α denotes learning rate (step-size) and the symbol γ denotes discount rate ($0 \leq \gamma \leq 1$). The discount rate determines how much the future rewards affect the state-action values. If $\gamma = 0$, the agent is only concerned with the immediate reward. As γ approaches 1, future immediate rewards have more impact on the cumulative reward.

The most obvious method of action selection (policy derivation) is to select the action with the highest Q -value. This method, called greedy policy selection, exploits the acquired knowledge to maximize the immediate reward, but it does not try seemingly inferior actions that may turn out to be more appropriate. An alternative to this purely exploiting approach is the ϵ -greedy policy selection. It acts as the greedy method most of the time, but once in a while, with small probability ϵ , it selects random action, independent of the current action values.

In a high risk application, such as drug dosing, choosing a purely random action may be undesirable. For this reason, we have been using greedy policy selection until now (Gaweda et al., 2005, Gaweda et al., 2007).

The algorithm presented above updates $Q(s, a)$ only for the action taken in the preceding step. If that action is penalized, a new action will be taken the next time the same state is encountered. However, this new action may not be better than the previous one. To illustrate this problem, let us consider a situation in which an rHuEPO dose leads to a Hgb decrease below the target range. Most dose-response characteristics are monotonically increasing. The Hgb vs. rHuEPO curve is no exception. In the described situation, trying a lower dose would not only slow the convergence to the optimal policy, but could also result in an undesirable Hgb levels (Gaweda et al., 2005).

To address this issue, we used a modified Q -learning algorithm (Gaweda et al. 2007) known in the literature as QS -learning, which stands for Q -learning with special spreading of action values (Ribeiro, 1998). In our instance of QS -learning, the state-action value was performed for every state-action pair using the following equation:

$$Q(s_i, a_i) \leftarrow Q(s_i, a_i) + \sigma(s, a, s') \alpha [r + \gamma \max_{a'} Q(s', a') - Q(s_i, a_i)]$$

The symbol $\sigma(s, a, s')$ denotes a spreading function. This function decides which state-action pairs are included in the update. We used an indicator function for $\sigma(s, a, s')$ with the following heuristics to select the updated state-action pairs:

- If $s < 11.5 \text{ g/dL}$ and $s' \leq s$, then $\sigma(s_i, a_i, s')=1$ for all $s_i \leq s$ and all $a_i \leq a$. If the last Hgb (s) is less than 11.5 g/dL and the new Hgb (s') is less than or equal the last Hgb, all state-action pairs for Hgb levels less than or equal the last Hgb and rHuEPO doses less than or equal the last dose must be updated.
- If $s = 11.5 \text{ g/dL}$ and $s' < s$, then $\sigma(s_i, a_i, s')=1$ for all $s_i \leq s$ and all $a_i \leq a$. If the last Hgb (s) is equal 11.5 g/dL and the new Hgb (s') is less than the last Hgb, all state-action pairs for Hgb levels less than or equal the last Hgb and rHuEPO doses less than or equal the last dose must be updated.
- If $s > 11.5 \text{ g/dL}$ and $s' \geq s$, then $\sigma(s_i, a_i, s')=1$ for all $s_i \geq s$ and all $a_i \geq a$. If the last Hgb (s) is greater than 11.5 g/dL and the new Hgb (s') is greater than or equal the last Hgb, all state-action pairs for Hgb levels greater than or equal the last Hgb and rHuEPO doses greater than or equal the last dose must be updated.
- If $s = 11.5 \text{ g/dL}$ and $s' > s$, then $\sigma(s_i, a_i, s')=1$ for all $s_i \geq s$ and all $a_i \geq a$. If the last Hgb (s) is equal 11.5 g/dL and the new Hgb (s') is greater than the last Hgb, all state-action pairs for Hgb levels greater than or equal the last Hgb and rHuEPO doses greater than or equal the last dose must be updated.

In the present work, we add the following heuristics to penalize all rHuEPO doses that lead to abrupt Hgb changes, which are biologically undesirable:

- If $|s - s'| \geq 2 \text{ g/dL}$, then $\sigma(s_i, a_i, s')=1$ for all $a_i \leq a$.
- If $|s - s'| \geq 2 \text{ g/dL}$, then $\sigma(s_i, a_i, s')=1$ for all $a_i \geq a$.

One of the distinct features of Reinforcement Learning especially with reference to supervised learning, is that the agent must learn the correct action on its own by exploring the environment. According to Thrun (1992), the exploration techniques can be generally classified into undirected and directed. Undirected methods explore the environment randomly, whereas directed methods utilize domain-specific knowledge to guide the exploration. Because of the random character, undirected exploration methods should generally be avoided in high risk applications. In the case of anemia treatment, for example, it would be undesirable to randomly explore too low rHuEPO doses if the Hgb is already far below the target range. Conversely, any exploration of high rHuEPO doses should be prevented if the Hgb is far beyond the target. To prevent the occurrence of such situations, we can use directed exploration.

With regards to our application, we need to address the spatio-temporal aspect of directed exploration, i.e. where and when to explore. Based on the a priori knowledge of the dose-response relationship

between Hgb and rHuEPO, we can establish the following heuristics. With the current state denoted by s , the exploiting (greedy) action by $a_{exploit}$, and the exploring action by $a_{explore}$:

- If $s < 11 \text{ g/dL}$, select $a_{explore} = (1 + \eta) a_{exploit}$ with probability ε .
- If $s > 12 \text{ g/dL}$, select $a_{explore} = (1 - \eta) a_{exploit}$ with probability ε .
- If $11 \leq s \leq 12 \text{ g/dL}$, select $a_{exploit}$.

The symbol η represents user-specified random variable defining the relative increase / decrease of the exploring action with respect to the exploiting action. In other words, for Hgb levels below the target range, we will try rHuEPO doses larger than the dose recommended by the greedy policy. Conversely, for Hgb levels above the target range, we will try rHuEPO doses smaller than the one recommended by the greedy policy. Finally, if the Hgb level is within the target, we will only perform exploitation. The proposed exploration scheme can be viewed as a directed ε -greedy approach. The complete algorithm can be summarized as follows.

```

Initialize state-action values  $Q(s_i, a_i)$ ,  $s_i \in S$ ,  $a_i \in A$ 
Initialize state  $s$ 
Repeat
    Derive policy  $\pi$  from  $Q(s_i, a_i)$ 
    Choose action  $a$  based on  $s$ :
        If  $s < 11 \vee s > 12$  then  $a = a_{explore}$  with probability  $\varepsilon$ 
        Otherwise  $a = a_{exploit}$ 
    Perform action  $a$ , observe next state  $s'$  and reward  $r$ 
    Update all state-action values:
        
$$Q(s_i, a_i) \leftarrow Q(s_i, a_i) + \sigma(s, a, s') \alpha [ r + \gamma \max_{a'} Q(s', a') - Q(s_i, a_i) ]$$

    Update state:
        
$$s \leftarrow s'$$

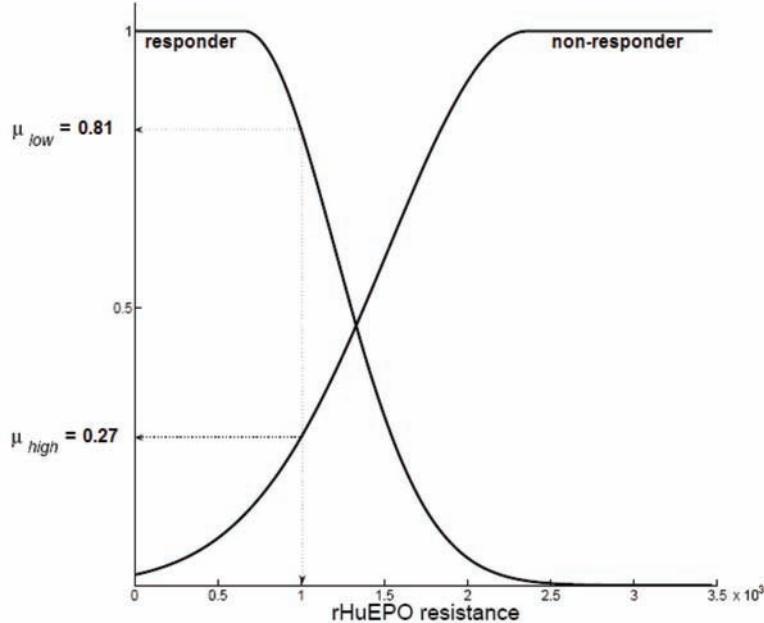

```

Experimental Results

To demonstrate the application of the proposed method to anemia management, we performed a computer simulation. For representing the dose-response relationship between rHuEPO and Hgb, we use a Takagi-Sugeno (TS) fuzzy model (Takagi & Sugeno, 1985). A TS fuzzy model can be viewed as a collection of local linear models with partial contribution toward the cumulative output. As a decision variable to determine the degrees of contribution of the local models, we use the rHuEPO resistance. The rHuEPO resistance is computed as a ratio of the three month average rHuEPO dose to the three month average Hgb level. Individuals with high rHuEPO resistance require large rHuEPO dose to achieve the target Hgb range, whereas low rHuEPO resistance means smaller rHuEPO dose may be sufficient. In other words, individuals with low rHuEPO resistance are more sensitive to rHuEPO than individuals with high rHuEPO resistance.

To build this model, we used data from 186 patients undergoing hemodialysis treatments at the Division of Nephrology of the Department of Medicine, University of Louisville. All of these patients received intravenous Epoetin Alpha which is the most commonly used isoform of rHuEPO in the

Figure 2. Fuzzy membership functions for the rHuEPO resistance.



United States. Using data records of 12 months, we calculated rHuEPO resistance for each individual. Using Fuzzy C-means clustering, we estimated fuzzy membership functions. Through experiments, we established that two clusters provided sufficient partitioning of the rHuEPO resistance domain. The two membership functions are shown in Figure 2. In this figure, individuals with predominantly low rHuEPO resistance are marked as “responders” and those with predominantly high rHuEPO resistance as “non-responders.” Figure 2 illustrates the case of an individual with rHuEPO resistance index of 1.0. This individual resembles a “responder” to degree 0.81 and a “non-responder” to degree 0.27.

The final model can be described in terms of TS fuzzy rules as follows:

R₁: IF patient type is responder

THEN $Hgb(k) = 0.72 Hgb(k-1) + 0.05 rHuEPO(k-1) - 0.006 TSat(k-1) + 3.21$

R₂: IF patient type is non-responder

THEN $Hgb(k) = 0.81 Hgb(k-1) + 0.03 rHuEPO(k-1) - 0.014 TSat(k-1) + 1.40$

The local linear models in rule consequents are Auto-Regressive with eXogenous inputs (ARX) models estimating Hgb one step (month) ahead, using past Hgb, past rHuEPO, and past TSat as regressors. For more details of this model the reader is referred to (Gaweda et al., 2007).

To perform the simulation of an ESRD patient population we used the above model in the following way. We first generated a set of 100 values for the rHuEPO resistance indices ranging from 100 to 3400. Each index corresponded to an individual patient and was kept constant throughout the simulation. We also generated a set of 100 TSat values ranging from 20 to 50 which corresponds to typical TSat values of Iron-replete patients most common within our ESRD patient population. In the clinical practice, the Iron repletion is achieved by maintaining the TSat level greater than 20%, administering external Iron

if necessary. In this simulation we focused only on the rHuEPO dose and assumed that the patients were Iron replete, i.e. no external Iron provided. Therefore, we set the TSat level constant per patient throughout the length of the simulation.

Following our previous work, we stored the state-action values in a look-up table. We established the state space S as a discrete set of Hgb levels between 9 and 14 g/dL with a step of 1 g/dL. Therefore, the rHuEPO dose associated with Hgb level is 9 g/dL would be administered for any Hgb level less than that. Conversely, the rHuEPO dose associated with Hgb level of 14 g/dL would be administered for any Hgb level greater than that. The action space A was a set of rHuEPO doses from 0 through 90,000 Units with a step of 2,500 Units. To compute the rHuEPO doses for the intermediate Hgb levels, we used linear interpolation as described in (Gaweda et al., 2007). The reward function used in the simulations had the following form:

$$r = -2(11.5 - \text{Hgb})^2$$

This reward achieves its maximum (0) if the Hgb reaches the median of the target range and exhibits a quadratic decrease as the Hgb departs the target range.

The Q -learning parameters were experimentally selected as follows:

- Initial learning rate $\alpha = 0.11$ and exponentially decreasing with time.
- Discount factor $\gamma = 0.99$ – emphasis on long-term rewards.
- Exploration probability $\varepsilon = 0.66$ – exploration is two times more likely than exploitation when allowed.

The exploring actions were allowed if either one of the following conditions was satisfied:

- If $Hgb < 11$ g/dL, then $rHuEPO_{explore} = (1 + \eta) rHuEPO_{exploit}$
- If $Hgb > 12$ g/dL, then $rHuEPO_{explore} = (1 - \eta) rHuEPO_{exploit}$

For η , we used a uniformly distributed random variable from 0 to 0.5. In other words, the exploring rHuEPO doses could be up to 50% higher or lower than their exploiting counterparts. The 50% margin is safe and consistent with our clinical protocol that allows up to 50% dose increase / decrease.

As we did previously in (Gaweda et al., 2007), we used the optimistic initial state-action values (Sutton & Barto, 1998), such that the initial policy would promote large rHuEPO doses for low Hgb values and small rHuEPO doses for high Hgb values.

The primary purpose of the simulation was to assess how quickly the Hgb level can be raised to the target range 11–12 g/dL within 12 months. Based on our patient population data, the baseline Hgb level was set to 8 g/dL and the initial rHuEPO dose was set to 0 Units per week. As a performance measure, we computed the Mean Squared Difference (MSD) between the achieved Hgb and the median of the target range (11.5 g/dL) over 12 months. We computed this metric for each one of the 100 simulated patients individually.

We benchmarked the tested algorithm against the following methods:

- Anemia Management Protocol (AMP) – the clinical protocol for rHuEPO dosing used at our dialysis unit.

Table 1. Simulation results

	AMP	<i>Q</i> -learning	<i>QS</i> -learning	<i>QS</i> -learning with smart exploration
MSD (mean \pm std dev)	1.96 \pm 0.89	1.43 \pm 1.04	1.31 \pm 1.02	0.70 \pm 0.62

- Standard *Q*-learning without exploration (Gaweda et al., 2005).
- *QS*-learning without exploration (Gaweda et al., 2007).

The simulations were performed in MATLAB® (The Mathworks, Natick, MA).

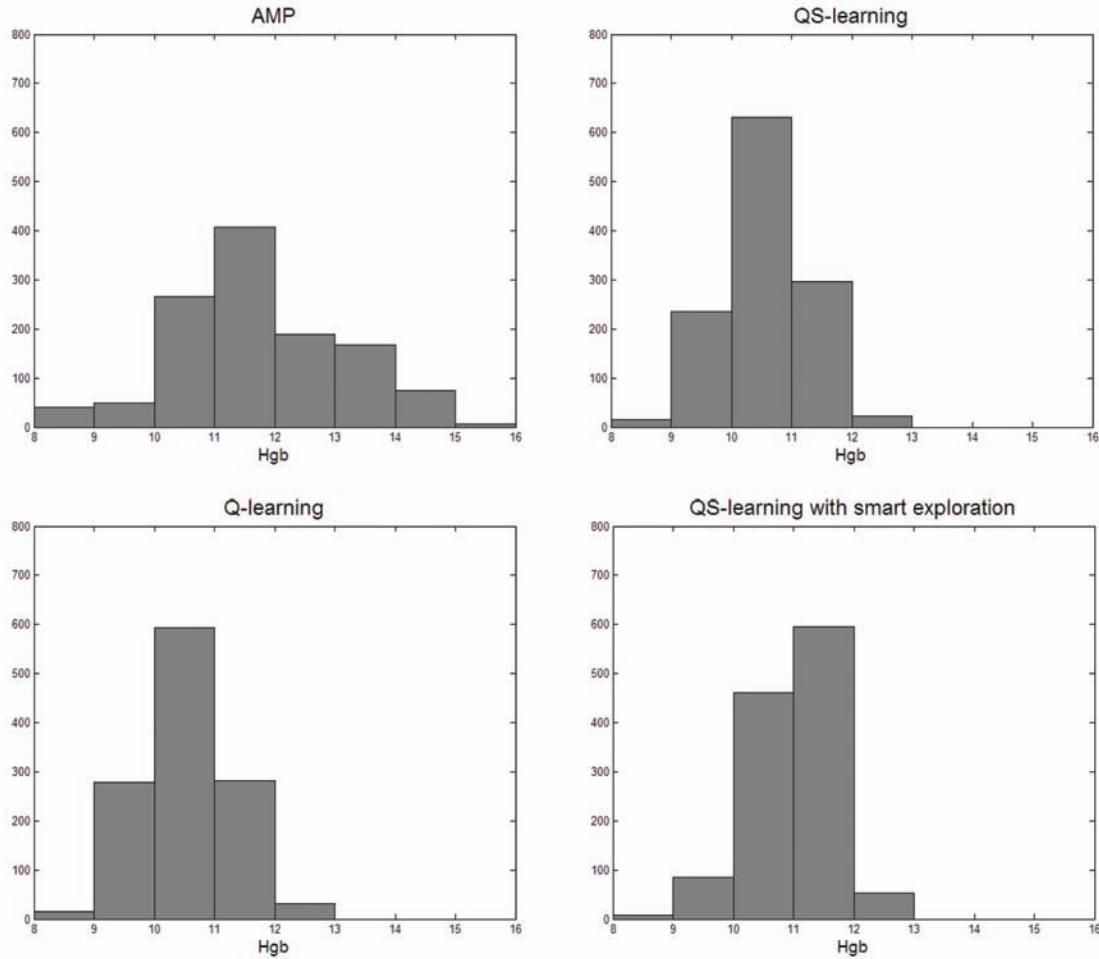
Table 1 summarizes the simulation results in terms of the mean values and standard deviation of the selected performance measure computer over all 100 simulated patients. The mean value defines how well the method is capable of achieving the treatment target on average. The standard deviation defines how consistent the method is in achieving the target across the simulated population. The most desirable scenario is one where both, the mean value and the standard deviation are small. Based on the results presented in Table 1, we can conclude that the proposed *QS*-learning with smart exploration outperforms the other methods. Remarkably, the mean MSD value for this method is almost two times smaller than that for the standard *Q*-learning. Figure 3 shows histograms of Hgb distributions as a result of simulated anemia treatment with each method. These histograms help determine the kind of Hgb response produced by each method. For example, the simulated AMP tends to overshoot the target range, which is consistent with our clinical experience. *Q*-learning shows a tendency to drive the Hgb toward the range of 10 to 11 g/dL instead of 11 to 12 g/dL. *QS*-learning seems to slightly improve upon *Q*-learning, but the improvement is significant only after the smart exploration component is incorporated.

Future Research

Determining a protocol for optimal drug administration in chronic illnesses is a time consuming process very often driven by trial and error. In this chapter, we proposed the use of Reinforcement Learning methods as a means to improve this process. Based on our previously reported results, we described a method that incorporates domain specific knowledge into the learning and exploration part. Although the presented results are very encouraging, the proposed method is by no means complete. So far, we have been developing and testing our methods under the assumption of time-invariant environment. However, it is possible patients may exhibit changes in their response to drug over time, for example, due to inflammation. Furthermore, Hgb level in patients may be a subject to external disturbances, such as blood loss due to internal bleeding. To fully address all the aspects of clinical care, further research efforts should focus on dealing with time-varying environments, as well as disturbance rejection.

The described approach by no means exhausts the possibilities of customizing standard Reinforcement Learning methods for the purpose of personalized drug dosing. One of the most interesting alternatives seems to be Reinforcement Learning with Apprenticeship proposed by Abbeel (2008).

Figure 3. Histograms of Hgb level distribution per method.



CONCLUSION

In this chapter, we have proposed the application of Reinforcement Learning to drug dosing personalization focusing specifically on the problem of anemia treatment with recombinant human erythropoietin. The main focus of our effort was the incorporation of the domain specific knowledge into the learning process to improve the speed of optimal policy search. The domain specific knowledge can be introduced into Reinforcement Learning in multiple ways. We have previously described an approach that external heuristics to guide the state-action value updates. In this chapter, we addressed the issue of exploration and proposed a method that performs directed “smart” exploration. The exploration is only allowed when the environment is not in one of the target states. Furthermore, the exploring actions depend on the state of the environment in relation to the target state. To evaluate this method, we have performed numerical simulations using a Takagi-Sugeno fuzzy model of an ESRD patient. These simulations show that the proposed method offers a significant performance improvement compared to a standard anemia treatment protocol, as well as a standard Reinforcement Learning method.

ACKNOWLEDGMENT

The work presented in this chapter has been supported by the National Institutes of Health under Grant, K25 DK072085.

REFERENCES

- Abbeel, P. (2008). *Apprenticeship learning and reinforcement learning with application to robotic control*. Unpublished doctoral dissertation, Stanford University.
- Dayan, P., & Balleine, B. W. (2002). Reward, motivation and reinforcement learning. *Neuron*, 36, 285–298. doi:10.1016/S0896-6273(02)00963-7
- Eschbach, J. W., & Adamson, J. W. (1985). Anemia of end-stage renal disease (ESRD). *Kidney International*, 28, 1–5. doi:10.1038/ki.1985.109
- Gaweda, A. E., Muezzinoglu, M. K., Aronoff, G. R., Jacobs, A. A., Zurada, J. M., & Brier, M. E. (2005). Individualization of pharmacological anemia management using reinforcement learning. *Neural Networks*, 18, 826–834. doi:10.1016/j.neunet.2005.06.020
- Gaweda, A. E., Muezzinoglu, M. K., Aronoff, G. R., Jacobs, A. A., Zurada, J. M., & Brier, M. E. (2007). Using clinical information in goal-oriented learning for anemia management. *IEEE Engineering in Medicine and Biology Magazine*, 26(2), 27–36. doi:10.1109/MEMB.2007.335580
- Harnett, J. D., Foley, R. N., Kent, G. M., Barre, P. E., Murray, D., & Parfrey, P. S. (1995). Congestive heart failure in dialysis patients: Prevalence, incidence, prognosis and risk factors. *Kidney International*, 47, 884–890. doi:10.1038/ki.1995.132
- Lacson, E., Ofsthun, N., & Lazarus, J. M. (2003). Effect of variability in anemia management on hemoglobin outcomes in ESRD. *American Journal of Kidney Diseases*, 41(1), 111–124. doi:10.1053/ajkd.2003.50030
- Lowrie, E. G., Ling, J., Lew, N. L., & Yiu, Y. (1994). The relative contribution of measured variables to death risk among hemodialysis patients. In E. A. Friedman (Ed.), *Death on hemodialysis: preventable or inevitable?* (pp. 121–141). Boston, MA: Kluwer Academic Publishers.
- Martin-Guerrero, J. D., Soria-Olivas, E., Chorro-Mari, V., Climente-Marti, M., & Jimenez-Torres, N. V. (2006). Reinforcement learning for anemia management in hemodialysis patients treated with erythropoietic stimulating factors. In *Proceedings of the 17th European Conference on Artificial Intelligence ECAI 2006*, Riva del Garda, Italy (pp. 19–24).
- Ribeiro, C. H. C. (1998). Embedding a priori knowledge in reinforcement learning. *Journal of Intelligent & Robotic Systems*, 21, 51–71. doi:10.1023/A:1007968115863
- Sinzinger, E. D., & Moore, B. (2005). Sedation of simulated ICU patients using reinforcement learning based control. *International Journal of Artificial Intelligence Tools*, 14(1-2), 137–156. doi:10.1142/S021821300500203X

- Sutton, R. S., & Barto, A. (1998). *Reinforcement learning. An introduction*. Cambridge, MA: The MIT Press.
- Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its application to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15, 116–132.
- Thrun, S. (1992). The role of exploration in learning control. In D. A. White & D. Sofge (Eds.), *Handbook of intelligent control: Neural, fuzzy, and adaptive approaches*. Florence, KY: Van Nostrand Reinhold.
- Watkins, C. J. C. H. (1989). Learning from delayed rewards. Unpublished doctoral dissertation, Cambridge University.
- Wolcott, D. L., Marsh, J. T., La Rue, A., Carr, C., & Nissensohn, A. R. (1989). Recombinant human erythropoietin treatment may improve quality of life and cognitive function in chronic hemodialysis patients. *American Journal of Kidney Diseases*, 14, 478–485.

KEY TERMS AND DEFINITIONS

Reinforcement Learning: area of machine learning concerned with how an agent ought to take actions in an environment so as to maximize some notion of long-term reward.

***Q*-learning:** Reinforcement Learning technique that works by learning an action-value function that gives the expected utility of taking a given action in a given state and following a fixed policy thereafter.

End Stage Renal Disease: Stage 5 of Chronic Kidney Disease synonymous with Glomerular Filtration Rate of less than 15 mL/min/1.73m² and requiring permanent renal replacement therapy (hemodialysis).

Anemia: qualitative or quantitative deficiency of hemoglobin, a molecule found inside red blood cells.

Erythropoietin: glycoprotein hormone that controls erythropoiesis, i.e. red blood cell production.

Chapter 13

Deterministic Pattern Mining on Genetic Sequences

Pedro Gabriel Ferreira

Centre for Genomic Regulation, Spain

Paulo Jorge Azevedo

Universidade do Minho, Portugal

ABSTRACT

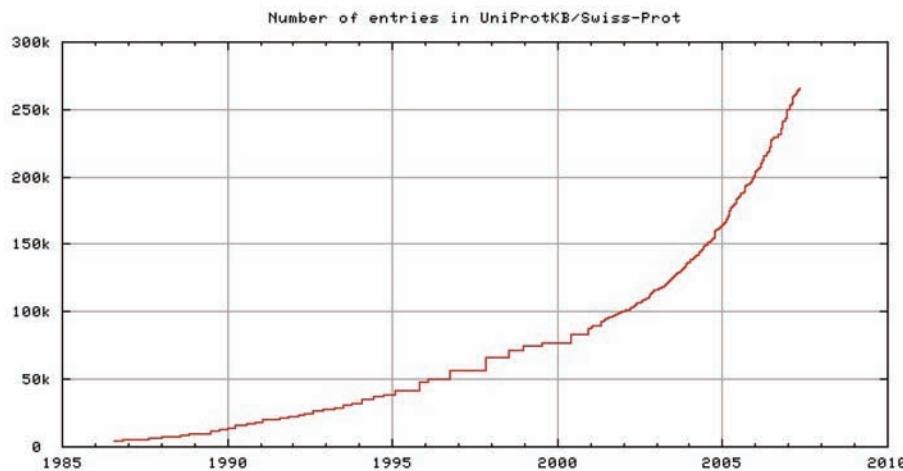
The recent increase in the number of complete genetic sequences freely available through specialized Internet databases presents big challenges for the research community. One such challenge is the efficient and effective search of sequence patterns, also known as motifs, among a set of related genetic sequences. Such patterns describe regions that may provide important insights about the structural and functional role of DNA and proteins. Two main classes can be considered: probabilistic patterns represent a model that simulates the sequences or part of the sequences under consideration and deterministic patterns that either match or not the input sequences. In this chapter a general overview of deterministic sequence mining over sets of genetic sequences is proposed. The authors formulate an architecture that divides the mining process workflow into a set of blocks. Each of these blocks is discussed individually.

INTRODUCTION

The Human Genome Project (NIH, 2007; Cooper, 1994) has led to the development of a set of new biological techniques. These new technological breakthroughs have in turn resulted on an exponential growth of biological data. In particular, the development of sequencing techniques culminated in a massive accumulation of genetic (DNA and protein) sequence data, which have then become freely available through specialized internet databases. For instance, the GenBank database contains approximately 85 759 586 764 nucleotides in 82 853 685 sequences (February 2008). The UniProtKB/Swiss-Prot contains 392 667 sequence entries, comprising 141 217 034 amino acids (July 2008). Figure 1 depicts the growth of UniProtKB/Swiss-Prot in the last years. Such amount of data raises big challenges both at the

DOI: 10.4018/978-1-60566-766-9.ch013

Figure 1. Growth in the number of protein sequences entered in the UniProtKB/Swiss-Prot database.



organizational and analysis level. In fact, the way that molecular biology research is done has changed significantly, being now a much more data-oriented and computationally based discipline. This fact, combined with the lack of robust theories to interpret these data opens a new and vast research area. The focus is now not on how to obtain the data, but on how to understand it. In the particular case of sequence data, this poses the question of how to retrieve biologically relevant patterns.

In the next two sections we briefly motivate the use of sequence mining to better understand DNA and protein mechanisms. More information about these topics can be easily found in any biology text book, for instance (Alberts, 2002; John, 1995). We also refer the reader to the following introductory articles (Hunter, 1993; Koonin, 2003; Brazma, 2001; Lesk, 2002).

DNA Sequence Mining

The DNA sequence contains all the necessary genetic information for the life of the being. Along this extremely long sequence, different regions encode different biological functional units. The function for some of these regions still needs to be determined. One of these regions is called *gene* and contains all the information necessary to create a protein. For a gene to be expressed, i.e. to result into a protein, it is necessary that a large number of conditions are fulfilled. One of these conditions is the existence of a certain types of sequences signals upstream and downstream the gene region. An important problem in bioinformatics is the identification of these signals in sequence segments found nearby genes. Such signals, called sequence patterns or motifs, are of major importance since they can provide insights into gene expression regulation.

Protein Sequence Mining

The analysis of a set of biologically related protein sequences may reveal regions of amino acid residues that occur highly conserved in several of those sequences. Such fact is certainly related to an evolutionary, chemical, structural or functional role of the protein. *Domains*, which consist of relatively small

structural and functional units, are an example of such regions. Capturing the characteristics of these regions through adequate pattern syntax, may reveal important insights about the proteins structural and functional role. Regular expressions provide a mechanism to efficiently search and select specific regions of a text. By making use of an enhanced regular expression syntax, *sequence patterns* or *motifs* are particularly good descriptors of regions with potential biological interest. They can, for instance, describe family *signatures*, i.e. subsequences that occur highly conserved in all or most of the sequences of a given family of proteins. Signatures can then be used to determine relations among the protein family in hand with yet uncharacterized proteins. Initially, the process of identification and description of signatures was performed by manual analysis. It was performed through the alignment of the target sequences, as was for example the case of the Prosite database (Hulo, 2007). With the growth of sequence data volume this approach is no longer feasible and efficient methods for the automatic retrieval of such patterns are required.

Different definitions of sequence pattern can be proposed according to the type of knowledge that is pursued. Two main classes of patterns can be distinguished: *probabilistic* and *deterministic*. Probabilistic patterns form a model that simulates the sequences or part of the sequences under consideration. When an input sequence is provided, a probability of being matched by the pattern is returned. Position Weight Matrices (PWM) and Hidden Markov Models (HMMs) are examples of probabilistic patterns. Deterministic patterns are commonly expressed through means of enhanced regular expression syntax, which can either match or not the input sequences. This chapter is devoted to analysis of deterministic sequence pattern discovery.

Scope of the Chapter

The work described in this chapter addresses the general problem of genetic sequence pattern mining: *how to efficiently discover sequence patterns conserved among a set of related genetic sequences*.

Sequence mining of protein and DNA databases represents two considerably different problems. The differences in the characteristics of the databases, like the length of the sequences, the size of the alphabet, the number of related sequences, similarity between sequences, requires the development of specific approaches for the efficient mining of each type of genetic sequence. Nevertheless, many aspects are common to the two approaches and methods exist that are efficient in both types of data. In this chapter we consider these two problems as a higher level general problem of genetic sequence mining. We review the entire process without making distinction and in section Methods we present methods both for DNA and protein sequence mining. In the context of this chapter the terms *sequence pattern* and *motif* have the same meaning and will be used interchangeably.

BASIC CONCEPTS

Two central concepts, regarding the discovery process, need to be defined: *search space* and *solution space*.

Definition 1 (Search and Solution space): Search space is the set of all possible sequence combinations that form the set of candidate patterns. Solution space corresponds to the set of sequence patterns that express the knowledge specified by the user.

A protein sequence consists of a string of amino acid symbols. The set of symbols is called the *alphabet* and denoted by Σ . Twenty different amino acids are typically considered:

$$\Sigma_{\text{Protein}} = \{\text{A, Q, L, S, R, E, K, T, N, G, M, W, D, H, F, Y, C, I, P, V}\}$$

The DNA alphabet is composed of four letters. Each letter is called a nucleotide base:

$$\Sigma_{\text{DNA}} = \{\text{A, C, G, T}\}$$

The length of a sequence S is denoted by $|S|$ and a set of sequences is represented as $I = \{S_1, S_2, \dots, S_k\}$.

In the context of sequence analysis the term *string* is often used as a synonym for sequence. However, the terms *substring* and *subsequence* are not equivalent. As defined in (Gusfield, 1999), a *substring* corresponds to a consecutive part of a string, while a *subsequence* corresponds to a new sequence obtained by eventually dropping some symbols from the original sequence and preserving the respective order. The later is a more general case than the former. Formally they can be defined as follows:

Definition 2 (Subsequence): For a sequence $S = s_1 s_2 \dots s_n$, \check{S} is a subsequence of S if $\check{S} = S_{i_1} \dots S_{i_m}$, with $i_1 < \dots < i_m$ and $m \leq n$.

Definition 3 (Substring): For a sequence $S = s_1 s_2 \dots s_n$, \hat{S} is a substring of S if $\hat{S} = s_{i+1} \dots s_{m+i}$, with $0 \leq i$ and $m+i \leq n$.

For the sequence $S = abcdef$, we have the subsequence $\check{S} = acdf$ and the substring $\hat{S} = cde$ as two possible examples.

Equivalently to Definition 3, we can say that a string y is a substring of S if exists strings x, z where $S = xyz$ with $|x| \geq 0$ and $|z| \geq 0$. x is said to be a *prefix* string of S and z a *suffix* string. In the same way, S is said to be a *superstring* or a *supersequence* of y . In section Output, more general definitions of subsequence and supersequence are presented.

When extending a sequence pattern $S = s_1 s_2 \dots s_n$, with a new symbol s_{n+1} , S is called a *base sequence* and $S' = s_1 s_2 \dots s_n s_{n+1}$ the *extended sequence*.

Different definitions of deterministic patterns have been proposed. In general a deterministic sequence pattern P can be defined as a non-null string of symbols from an enhanced alphabet Σ' . The alphabet Σ' results from an extension of Σ with an additional set of symbols E , $\Sigma' = \Sigma \cap E$.

E may contemplate symbols that express the existence of gaps or that certain positions may be occupied by more than one symbol. This extension adds greater expressiveness to the patterns. The set of all the possible regular expressions that results from the substitution of symbols from Σ' in P defines a *pattern language* $L(P)$. In section Output, an enumeration of different pattern languages and the respective type of patterns is made.

Definition 4 (Match): A pattern P is said to match a sequence S , if S contains some string y that occurs in $L(P)$.

Definition 5 (Support): The number of times a pattern P occurs in the input database I , is called support of P and denoted as σ .

The support can be expressed in terms of the number of matches of P in different sequences or in the total number of matches of P in the database. In the second case, each sequence may contribute more than once to the count of the support. Depending on the problem being tackled and on the method being used, a proper definition is chosen. The *cover* of the pattern represents the list of sequence identifiers where the pattern occurs.

Definition 6 (Frequent Pattern): A pattern P is said to be frequent if it occurs a number of times greater than the minimum support threshold; and is infrequent otherwise.

The statistics from UniProtKB/Swiss-Prot (Hulo, 2006) indicates that the average protein sequence length is 365, with a maximum length reaching more than 34 000 and minimum length of 5 amino acids. This relatively long length combined with a small alphabet, makes protein sequences a very dense combination of amino acids. Indeed, the number of possible random amino acid sequences is extremely large. For a sequence of length L , there are $|\Sigma|^L$ possible sequences. Looking for sequence patterns in such a huge space demands efficient mechanisms and techniques.

One possible way to tackle the complexity of sequence pattern discovery is by introducing domain knowledge that the analyst/user knows about the problem. This knowledge is expressed through means of constraints. When incorporated into the mining process, the constraints lead to a possible reduction on the search space and therefore allow a more efficient enumeration of the solution space. The minimum support is the first of the constraints to be applied. Patterns occurring less than the user defined minimum support value are discarded. Other constraints and features used to specify the characteristics of the reported patterns will be introduced in section Parameters. We follow by defining the problem associated to sequence pattern discovery:

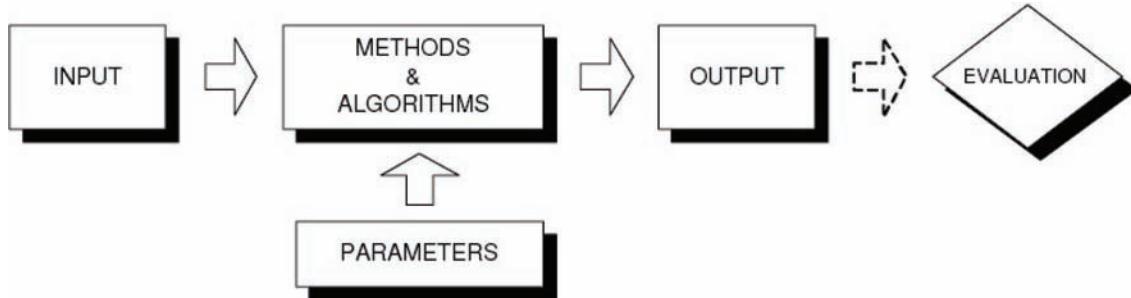
Definition 7 (Sequence Pattern Discovery Problem): Given an input a set of genetic sequences $I = \{S_1, S_2, \dots, S_k\}$, defined over Σ^* , a minimum support value σ ($\sigma > 0$ and $\sigma \leq k$) and an additional set of parameters defining other constraints, find all the sequence patterns in I with a support greater or equal than σ , fulfilling the specified constraints.

PATTERN DISCOVERY WORKFLOW

Sequence pattern discovery is an active area of research for more than fifteen years. During this time many different computational methods have been proposed. Many of these methods show completely different algorithmic approaches to the problem, ranging from the format of the input data, the type of extracted patterns and the computational techniques applied. A considerable amount of novel approaches are focused on some specific datasets or problems. This makes performance and accuracy comparisons hard to attain.

Identifying the requirements, limitations and outcomes of a certain method can sometimes be an intricate task. Nevertheless, some components are common to all of these approaches. Figure 2 depicts a generic architecture for a sequence pattern discovery method. The “Input” block specifies the format of the input sequences. The “Output” block specifies the type and the features of the extracted sequence patterns. The “Algorithms and Methods” component describes the computational techniques applied to extract all the patterns. The “Parameters” block is used to set all the variables required by the method

Figure 2. Architecture of Sequence Pattern Discovery Methods.



and the user/analyst. The “Evaluation” block is an optional component since the assessment of the significance of patterns frequently appears as a post-processing step.

In the next section, each of these blocks is described in detail. Additional concepts and definitions will be introduced and according to this, references to the different algorithms and methods will be provided.

Input

Multiple sequence alignment consists of identifying regions of similarity across several sequences (Gusfield, 1999). Two types of alignments are possible: *global* and *local*. Global alignments try to align of a maximum number of amino acid residues positions. Local alignments point out relatively small and well conserved regions that may occur in dissimilar sequences. Figure 3 shows an example of these two types of alignments. These regions may result as a consequence of a functional, structural, or evolutionary relationship. If pattern syntax is used then these regions can be expressed as sequence patterns.

From this, it can be identified that there is a tight relation between multiple sequence alignment and pattern discovery methods. The former highlights conserved regions that when analyzed and processed can be used to express sequence patterns. An example of such an algorithm is the *eMotif* (Huang, 2001; Wu, 1995). It assumes that the input is a multiple sequence alignment, where the alignment is used to guide the recursive enumeration of multiple sequence patterns that fulfill the user requirements. This type of approach is only feasible when sequences share a considerable degree of similarity. When this is not the case, no significant alignment is obtained and therefore no conserved regions are detected. Additionally, multiple sequence alignment algorithms are computationally intensive. The search for the optimal solution is a NP-hard problem (Wang, 1994). Heuristics need to be introduced and therefore only a suboptimal solution is guaranteed. This type of methods finds only a small subset of the frequent patterns, which occur in the majority of the sequences. Patterns occurring only in subsets of the input sequences are much more difficult to detect and usually not reported.

If sequence patterns are known in advance, they can be used as reference points to align multiple sequences. The algorithm *MUSCA* (Parida, 1999) is a two phase algorithm that in the first phase finds sequence patterns that occur among subsets of the input sequences. In the second phase, these patterns are used to produce a multiple alignment.

Algorithms for sequence pattern discovery can be categorized according to the format required for the input sequences (Brazma, 1998a). Sequence alignment algorithms require a multiple sequence alignment

Figure 3. Simple example of global and local alignment between two sequences.

GLOBAL: TAGATCGGCCATA
 | | | | | | | |
 AG -- CGG - - TA (7)

LOCAL: TAGATCGGCCATA
 | | | | | | |
 AGCGGTA (5)

as input. Pattern enumeration algorithms use a more general approach to the problem and therefore are able to handle unaligned sequences. In this chapter we will focus on the latter category of algorithms.

Output

Deterministic sequence patterns can be divided into two types: *Fixed-length* and *Extensible-length*. We introduce now some concepts useful for characterizing sequence patterns.

Definition 8 (String Distance): Given two strings x and y , where $|x| = |y|$, the distance (dissimilarity) between them is given by the Hamming distance $H(x, y)$, i.e. by the number of mismatched symbols.

For instance, $x=ACDEF$ and $y=ABDET$, $H(x, y)=H(y, x)=2$.

Definition 9 (String Neighborhood): Given a string x , all the strings that can be found at a Hamming distance of at most d symbols from x are said to be in the d -neighborhood of x .

Definition 10 (Fixed-length patterns): Fixed-length patterns, also known as (l, d) -motifs (Sagot, 98; Pevzner, 2000; Buhler, 2001), consist of sequence patterns of a fixed size of l symbols, where the matched substrings in the input database are in its d -neighborhood.

Generally the problem of finding (l, d) -motifs, can be defined as follows:

Definition 11 (Planted (l, d)-motif Problem): Find the unknown motif M of a fixed size l , in a set of t input sequences of length n , which it is known to have exactly one occurrence Q in each sequence, such that $H(Q, M) = d$.

The motif M is said to be a *model*, since it is in the center of the d -neighborhood defined by all the Q occurrences. Changing d positions in M implies deriving all Q strings. Two notes should be made regarding this type of patterns. First, it could be the case that all Q occurrences of M can be found, but the model M itself may never be found. M has then to be determined from the set of all the strings Q_1, Q_2, \dots, Q_t in what is called the Steiner sequence problem (Lonardi, 2002).

The second point refers to the fact that any two occurrences of M differ at most $2d$ positions. For larger values of l , larger values of d are usually allowed. This can be problematic since two occurrences of M may distance by a large number of positions.

The planted (l,d)-motif problem was formalized by Pevzner and Sze (Pevzner, 2000). A particular version of this problem, named as the “*challenge problem*”, is to find a planted (15, 4)-motif in a set of 20 DNA sequences of length 600.

Extensible-length patterns have an arbitrary length and are expressed through enhanced regular expressions. Depending on the applications that generate the patterns, different versions of regular expression syntax can be used. A generic format for these types of patterns is defined as follows:

$$A_1-x(p_1,q_1)-A_2-x(p_2,q_2)-\dots-A_n \quad (1);$$

where A_k is a sequence of consecutive symbols, and $-x(p_i, q_i)$ - a gap greater than p_i and smaller than q_i . A gap corresponds to positions that can be matched by any symbol. If p_i equals to q_i , the gap notation is usually abbreviated to $-x(p_i)$ - . If a position is occupied with more than one symbol, a bracket notation is used. For instance, the notation $[ACG]$ denotes a position that can be occupied by the symbols A , C or G .

The Prosite database (Hulo, 2006) is one such example that uses a regular expression syntax to express their patterns. The pattern $C-x-C-x(2)-[GP]-[FYW]-x(4,8)-C$ corresponds to the entry PS01186. This describes a pattern where the first position can be occupied by the amino acid Cysteine (C), followed any amino acid (don't care position), by another Cysteine (C) and by any two amino acids. The sixth position can be occupied by a Glycine (G) or Proline(P) and the seventh by a Phenylalanine (F) or Tyrosine (Y) or Tryptophan (W). A gap of size four to eight occurs before the last position that is a Cysteine (C). A “don't care” position can also be expressed through the symbol ‘.’. The above pattern can be rewritten $C . C . [GP]-[FYW]-x(4,8)-C$.

Deterministic sequence patterns have a great expressive power, being good descriptors for well conserved regions across several protein sequences. Furthermore, since they are not confined to a fixed length, they can indicate relations among distant related amino acids. Four types of deterministic sequence patterns can be distinguished, according to the pattern language to which they belong:

- *Concrete Patterns* admit only concrete contiguous symbols, i.e. no gaps are allowed and all the symbols are in Σ . These patterns are defined over the Σ^* language, which is the simplest pattern language. Each position of the pattern is undoubtedly occupied by the respective amino acids symbol. Ex: C-D-G-P-G-R-G-G-T-C (Prosite entry PS00419).
- *Ambiguous Patterns* only contain contiguous symbols, but some positions may be occupied by more than one symbol. This syntax is defined by the language $(\Sigma \cap R)^*$, where $R = \{R_1, \dots, R_n\}$ is a collection of sets $R_i \in \Sigma$. Each set R_i expresses the amino acids symbols that can be found in an ambiguous position. Ex: V-[DN]-Y-[EQD]-F-V-[DN]-C (Prosite entry PS00772).
- *Rigid Gap Patterns* only contain gaps with a fixed length; this corresponds to a case where $p_i = q_i, \forall i$ (see Formula (1)). The defined pattern language is $(\Sigma \cap \cdot)^*$, where ‘.’ denotes a position that can be occupied by any amino acid. Ex: Y..Y-Y.C.C (Prosite entry PS00121).
- *Flexible Gap Patterns* allow gaps of variable length; the respective pattern language is $(\Sigma \cap G)^*$, where G is a component on the form $-x(p_i, q_i)$ - . It corresponds to a case where $p_i \leq q_i, \forall i$ (see Formula (1)). Ex: [RK]-x(2,3)-[DE]-x(2,3)-Y (Prosite entry PS00007).

In general, it is assumed that a sequence pattern starts and ends with a concrete or ambiguous position, i.e. with symbols from Σ or $\Sigma \cap R$ (R is defined as for the ambiguous patterns). One should note

that (l, d) -motifs are a particular case of ambiguous patterns of fixed size.

The operations performed over extensible-length deterministic sequence patterns can be summarized in three types:

Definition 12 (Specialization): The operation of replacing a “don’t care” symbol by a concrete symbol is called *specialization*.

If M' is specialization of M , then $\text{support}(M) \geq \text{support}(M')$. Ex: C . CL . A is a specialization of C . C . A.

Definition 13 (Generalization): The substitution of a concrete symbol by an ambiguous position or by a “don’t care” is called *generalization*.

If M' is generalization of M , then $\text{support}(M) \leq \text{support}(M')$. Ex: C . . . A is a generalization of C . C . A.

Definition 14 (Extension): Append one or more concrete symbols or ambiguous positions to the left or the right of a pattern is called an *extension*.

If M' is an extension of M , then $\text{support}(M) \geq \text{support}(M')$. Ex: C . CL . A [LY] is an extension of C . CL . A.

A set of well defined properties that makes its biological characteristics unique is assigned to each AAs. However, some of these properties are common to different types of the amino acid. This enables to group amino acids according to their chemical and structural properties. These groups are typically called *substitution sets*. A common observation is the replacement of some amino acid by another, without lost of pattern function (Henikoff, 1993). One such example is the Prosite pattern: [RK]-x(2,3)-[DE]-x(2,3)-Y (entry PS00007). These substitutions can be expressed through ambiguous positions.

Definition 15 (Pattern Containment): A sequence pattern M' is said to be *contained* in a pattern M , and is called *subsequence* of M , if it can be obtained by dropping some symbols from M or by a specialization of M . This is denoted as $M \sqsubseteq M'$.

For instance, A . . C . L is a subsequence of A . . C . LT; A . . C . L is a subsequence of A L. Similarly, M contains M' and is called *supersequence* of M' , if M can be obtained by an extension or a generalization of M' .

A sequence pattern can be classified into three different classes: *All*, *Closed* and *Maximal*. Considering the set of the frequent sequence patterns FS :

Definition 16 (All): A pattern M belongs to the *All* class (Brazma, 1998a; Wu, 1995) if it is a frequent sequence pattern, i.e. $M \subset FS$.

Definition 17 (Closed): A sequence pattern M is called *Closed* (Rigoutsos 1998; Califano, 2000) when all its extensions and specializations result in sequence patterns with smaller support, i.e. $\nexists M' \in FS$ such that $M \sqsubseteq M'$ and $\text{support}(M) = \text{support}(M')$.

Definition 18 (Maximal): A sequence pattern M is said to be *Maximal* (Ester, 2004) when it is frequent and it is not contained in any other sequence pattern, i.e. $\nexists M' \in FS$ such that $M \sqsubseteq M'$.

The All class corresponds to the full set of frequent patterns. Maximal patterns are a subset of closed

Figure 4. Sequence database and respective patterns.

Database	Rigid Gap Patterns		Flexible Gap Patterns
	A,G,C,D,E	AG (2), A.C (2), A...E (2), C.E (2), CD (2)	A,G,C,D,E A-x(2,3)-C (2) A-x(1,2)-C (3)
AGYCD AGCDE AFCKE Min Support = 2/3 of sequences	A.C.E (2)		A-x(1,2)-CD (2)

patterns, which are a subset of the All class. Figure 4 shows an example of sequence database and the respective rigid and flexible gap patterns obtained for a minimum support of $2/3$ of the number of sequences. Patterns are enumerated, and the respective support provided. It is shown in different levels according to the number of concrete symbols that they contain. Patterns A . C . E and A-x(1,2)-CD are maximal since they are not contained in any other pattern. A-x(1,2)-C is closed since all its extensions have a smaller support.

Evaluation

Depending on the setting used in the mining process, a large amount of frequent patterns may be reported. This number is a direct outcome of the user defined parameter values but also of the properties of the dataset and the algorithm characteristics. Some of these patterns arise by chance and have no biological interest. It is critical to evaluate the significance and the interest of the reported patterns. Statistical and biological relevance are often correlated. Therefore, ranking patterns according to statistical or information gain relevance enables an easier interpretation of the results. This evaluation is usually separated from the mining process and appears as a post-processing step.

As formalized by Brazma et al. in (Brazma, 1998), a significance measure can be defined as function in the form: $f(m, C) \rightarrow R$. m stands for the pattern being evaluated and C a set of related sequences where the pattern is expected to have some biological significance. This function returns a real value score that expresses how relevant or significant is m with relation to C .

These scores may provide hints to evaluate biological significance. If additional sequence information is available, where patterns are expected to have a bad matching performance, the evaluation can consider both positive and negative information. Four possible cases of a pattern m matching a sequence of C can be distinguished:

- *True Positives (T_p)* - number of sequences that belong to the target family and match the pattern.
- *True Negatives (T_N)* - number of sequences that do not belong to the target family and do not match the pattern.
- *False Negatives (F_N)* - number of sequences that belong to the target family and do not match the pattern.

Figure 5. Table with example of significance measures typically used in machine learning and motif mining algorithms.

Symbol	Measure	Formula	Range
Sn	Sensitivity	$Sn(M) = \frac{TP}{TP+FN}$	[0,1]
Sp	Specificity	$Sp(M) = \frac{TN}{TN+FP}$	[0,1]
PPV	Positive Predicted Value	$PPV(M) = \frac{TP}{TP+FP}$	[0,1]
Fpr	False Positive Rate	$Fpr(M) = \frac{FP}{FP+TN}$	[-1,1]
F	F-Measure	$F(M) = \frac{2 \times Sensitivity \times PPV}{Sensitivity + PPV} = \frac{2 \times TP}{2 \times TP + FN + FP}$	[0,1]
Corr	Correlation	$C(M) = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FN)(TP+FP)(TN+FP)(TN+FN)}}$	[-1,1]
IG	Information Gain	$IG(M) = I(M) \times [Support(M) - 1]$ where $I(M) = -\log_{ \Sigma } P(M)$	[0, +∞[
Pratt	Pratt Measure	$Pratt(M) = \sum_i^n I'(A_i) - c \cdot \sum_{k=1}^{n-1} (q_k - p_k)$ where $I'(A_i) = -\sum_{a_i \in A_i} (P(a_i) \times \log(P(a_i))) + \sum_{a_i \in A_i} (\frac{P(a_i)}{P(A_i)} \times \log(\frac{P(a_i)}{P(A_i)}))$ and $P(A_i) = \sum_{a_i \in A_i} p(a_i)$] - ∞, +∞[
LogOdd	LogOdd	$\text{LogOdd}(M) = \log(\frac{\text{Support}(M)}{P(M)})$] - ∞, +∞[
ZScore	Z-Score	$ZScore(M) = \frac{\text{Support}(M) - E(M)}{N(M)}$ where $E(M) = N_{\text{resid}} \times P(M)$ and $N(M) = \sqrt{N_{\text{resid}} \times P(M) \times (1 - P(M))}$] - ∞, +∞[

- *False Positives (F_p)* - number of sequences that do not belong to target family and match the pattern.

Figure 5 presents a table with some measures that have been used for motif evaluation. The first group: Sensitivity, Specificity, Positive Predicted Value, False Positive Rate, F-measure and Correlation are measures well known in the machine learning field and can be find in any book on this topic (e.g. (Witten, 2005; Han, 2001)). These measures have also been widely applied in bioinformatics (Baldi, 2000). The second group of measures has been applied with particular success in sequence mining. Information Gain has been applied in the eMotif algorithm (Wu, 1995), Pratt measure in (Jonassen, 1995), LogOdds in (Durbin, 1998; Krogh, 1998) and Z-Score in many other methods like for instance Verbumculus (Apostolico, 2003; Apostolico, 2005). Studies on the evaluation of a set of measures in the presence of different pattern and dataset characteristics can be found in (Lin, 2006; Ferreira, 2007).

Parameters

Sequence mining tends to derive too many patterns, most of them useless or redundant. The characteristics of the input database and mining methods help in explaining this phenomenon. This is especially true when databases attain considerable size or when the average length of the sequences is very long, a typical scenario in genetic sequence data. In such cases the mining process becomes computationally expensive or simply infeasible. As described in section BASIC CONCEPTS, the minimum support threshold is the first constraint to the mining process imposed by the user. This parameter has a direct impact in the size of the solution space as it imposes a confinement on the search space. The higher the minimum support, the smaller the number of frequent patterns. The support of a pattern can be considered the first measure of interest.

Additionally to the minimum support, other user constraints appear as an efficient way to prune the search space and to help the search method in focusing on the expected patterns. The use of constraints enhances the execution of a database query. The runtime reduction grants the user the opportunity to

interactively refine the query specification. This can be done until a satisfactory answer is found. The most common and generic types of constraints are:

- Item Constraints: defines the set of the events (*excludedEventsSet*) that may not appear in the pattern.
- Gap Constraints: defines the minimum (*minGap*) or the maximum (*maxGap*) distance that may occur between two adjacent events in the sequence patterns.
- Gap Penalty Constraints: measures the density of a pattern, through the ratio between the number of concrete events and the span of the pattern.
- Duration or Window Constraints: defines the maximum distance (*window*) between the first and the last event of the sequence pattern.
- Start Events Constraints: determines that the extracted patterns start with the specified events (*startEvents*).

Methods

The size of the search space is proportional to the expressive power of the pattern language, i.e. the number of possible patterns that can be derived from the selected language. Consider the simplest pattern language, Σ^* . For a length L , $|\Sigma|^L$ concrete patterns are possible. If the alphabet is extended with the “don’t care” symbol then $(|\Sigma|+1)^L$ possibilities exist. The size of the solution space is expected to be a direct outcome of the characteristics of the input sequences, namely its length, number and the restrictiveness of the user defined parameters (minimum support and constraints).

The aim of a sequence pattern discovery algorithm is to traverse the search space in an efficient way concerning time and space. Regarding the way the solution space is enumerated, in particular the number of reported patterns, an algorithm is said to follow one of two approaches: *complete* or *sound* approach (Lonardi, 2002). In the complete approach, the goal is to find all possible patterns. These algorithms can also be called “combinatorial” or “exact” (Floratos, 1999). Since too many patterns may arise, a second type of algorithms follow a sound approach, only guaranteeing that no false patterns (in the sense of their interest/significance) are reported.

Approximation techniques can be employed to improve the efficiency and quality of the mining process. The use of such heuristics will eventually lead to a solution where a small set of high quality patterns are reported but no guarantee of finding the optimal solution is given. These algorithms are commonly denominated as “heuristic” based algorithms.

In order to efficiently traverse the search space and enumerate the frequent patterns, several techniques can be employed and combined. The existence of common techniques and the type of extracted patterns leads to a categorization of the algorithms. The following topics, in particular the type of extracted patterns, will be used as the major guidelines for obtaining this categorization:

- Traversal direction: bottom-up or top-down.
- Traversal mode: depth-first or breadth-first.
- Class of extracted patterns: all, closed or maximal.
- Type of extracted patterns: concrete, ambiguous, rigid-gap and flexible-gap.
- Heuristics: applied or not.

In the following section, a survey of several sequence pattern discovery algorithms is described. The algorithms here presented were selected as examples to illustrate the approaches relative to each of the above topics. Therefore this discussion is certainly not exhaustive. A complete review of motif mining algorithms can be found for instance in (Sandve, 2006; Das, 2007; Tompa, 2005).

Exhaustive Search

Concrete patterns are particularly suitable to be extracted through an exhaustive search approach. This is the simplest pattern discovery approach and can be summarized in four steps (Lonardi, 2002). The user starts by defining the search space, which in this case consists of the set of concrete patterns, and the respective constraints. Next, an exhaustive enumeration of all the patterns from search space is made, which typically leads to the generation of a large number of patterns. In the third step the computing of the support and significance of each sequence pattern is performed. In the last step, patterns that fulfill the user constraints and with the highest score (or with a score above the defined threshold) are reported.

The exhaustive enumeration has the advantage of being a simple and exact approach, where the best patterns are guaranteed to be found. A major drawback lies in the fact that its time complexity is exponential with the length L of the extracted patterns ($O(|\Sigma|^L)$) resulting in an approach only suitable to discover sequence patterns with small length. Several algorithms have been proposed to overcome this inefficiency.

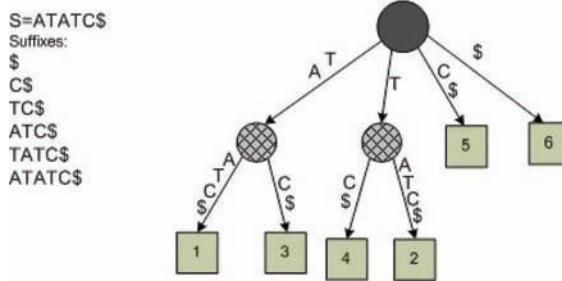
Verbumculus (Apostolico, 2003; Apostolico, 2005) and *Weeder* (Pavesi, 2001) are two examples of such algorithms. Both make use of an efficient data structure called Suffix Tree (Weiner, 1973; McCreight, 1976; Ukkonen, 1995). A suffix tree is a tree that stores all the suffixes of a string. It is designed to allow a fast string lookup. Verbumculus was designed to extract concrete patterns and Weeder to extract (l,d)-motifs, where the number of mismatches, d , is proportional to the length l . A suffix tree T of a string S with length N , is a rooted tree with the following characteristics (Weiner, 1973):

- it has exactly N leaves;
- each internal node other than the root has at least two children;
- each edge is labeled by a substring of S ;
- no two edge labels out of a node start with the same character;
- for any leaf i , the concatenation of the edge labels on the path from the root to i spells out the suffix $x_i \dots x_m$.

The set of sequences from the input dataset are concatenated and separated by a special marker symbol (it does not appear in any of these sequences). A special end character is also added. This string is then used to construct the tree. Figure 6 shows an example of a suffix tree for a small string. A suffix tree for a set of sequences is often called *Generalized Suffix Tree*. The construction of this data structure takes $O(N)$ time and $O(N)$ space (Weiner, 1973; McCreight, 1976; Ukkonen, 1995), where N is the sum of the length of all the strings. It is then annotated by a k -bit string, where the i -th bit is set if the corresponding string is found in the i -th sequence. This allows for a fast support checking. The annotated suffix tree will contain a path for each string occurrence in the input set of sequences.

Based on this property two major operations can be performed over suffix trees: *traversals* and *queries*. Traversals carry out an enumeration of the patterns present in the input dataset. They can be performed level-by-level in a breadth-first mode or they can be done in depth-first manner. These two

Figure 6. Suffix tree for the string ATATC with end marker \$. Each leaf has the index of respective string.



modes will be discussed later in this chapter. They can also be used to check for the existence of a given pattern. For a pattern of length m this operation takes $O(m)$ time.

Verbumculus makes use of the traversal operation of an annotated suffix tree to obtain all the patterns that occur in the input dataset. Patterns that occurs an “unusual” number of times are reported. A pattern is said to be unusual if it is over or under represented. This can be measured through the use of a statistical function, like for example the Z-Score function. This measure and its monotonic properties are discussed in (Apostolico, 2005). Suffix trees have proven to be the most efficient technique to extract concrete patterns (Apostolico, 2003).

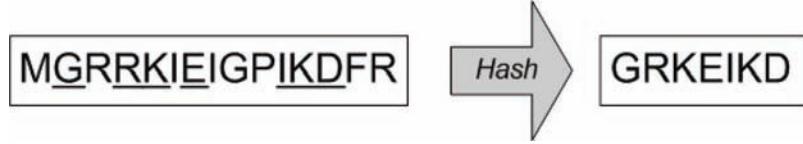
Biological sequences show a considerable complexity. For instance, in different protein sequences, regions with an analogous function may have a significantly amino acid composition. This is so, because mutations, deletions and insertions can occur. Due to its limited expressiveness, concrete patterns cannot be used to express patterns that do not occur exactly. Ambiguous patterns overcome part of this limitation by handling symbol mutations.

A large research effort has been done in developing methods for efficiently extracting (l,d) -motifs. Three of these methods and the respective computational approaches are now briefly discussed. They are presented following a historical sequence.

Exhaustive Graph Search

This type of algorithms is based on the idea of searching by substring combinations of the input sequences that can be possible instances of a motif. *Winnower* (Pevzner, 2000) is an example of such an algorithm. It builds a graph where each vertex of the graph corresponds to a substring of length l . Two vertices are connected by an edge if the Hamming distance between the respective substrings is no more than $2d$ symbols and they do not come from the same input sequence. This forms a n -partite graph, which means that the graph can be partitioned in n partitions provided that there is no edge between the vertices in a partition. Each partition corresponds to one of the input sequences. A *clique* is a set of n vertices such that two vertices are connected by an edge. The goal is then to find all cliques that have exactly one vertex in each partition. Since looking for large cliques in graphs is a NP-complete problem, pruning strategies are introduced in order to reduce the number of edges that cannot be part of any clique of size n . The reduction in the number of edges lowers the complexity of finding cliques. Winnower uses the notion of an expandable clique, which consists of a clique that contains at least one vertex in a different partition of the n -partite graph. A vertex is called a *neighbor* if it is connected by an edge to each vertex of the

Figure 7. Hash example for a random projection.



clique. The algorithm starts with a subset of k vertices, and finds all the expandable cliques of size n . There are $\binom{n}{k}$ possible expandable cliques. The pruning strategy consists of removing edges that are not in $\binom{n-2}{k-2}$ of these cliques. This is a random approach since by removing these edges some of the cliques may be missed. Consequently, it is necessary to make several iterations in order to ensure completeness. The larger the value of n , and consequently the value of k , the longer it takes. The Winnower algorithm is based upon the idea of finding cliques with exactly one vertex per partition. It assumes that a pattern occurs exactly once per sequence, which is a condition too rigid for most of the cases.

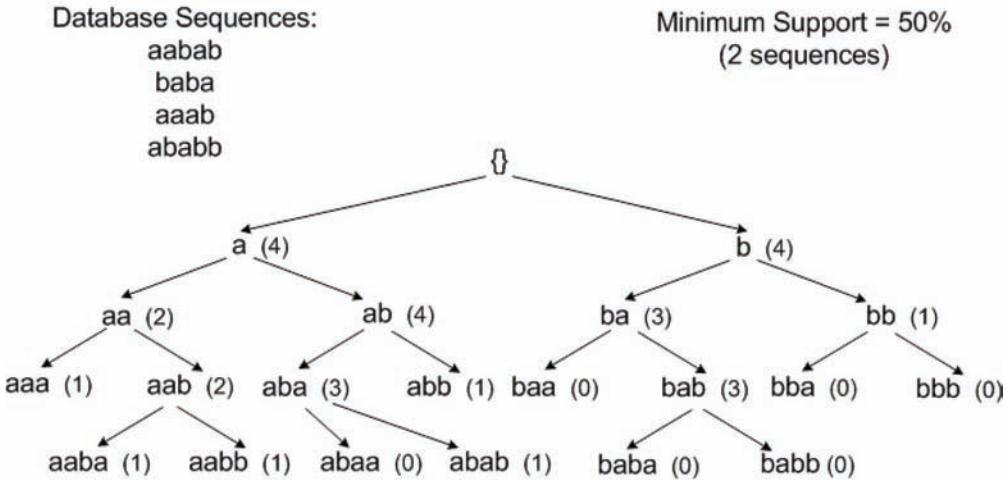
Random Projection Search

The *Random Projection* algorithm, proposed by Buhler and Tompa (Buhler, 2001), was designed to tackle the (l, d) -motif problem. It is based on the projection idea, which can be described as the process of hashing substrings of the input sequences. For each substring of size l a key is generated. This key consists of a gapped pattern obtained through the random selection of k symbols of the substring, as exemplified in Figure 7. It is required that $k \leq l-d$ in order to avoid including mutated symbols in the key. A hash table is used to store all the cells and the respective support. This process is repeated to all the substrings several times (with different hash functions). Over represented cells on the hash table are expected to correspond to motifs instances. A post-processing step uses the high scoring cells to cluster instances of motifs. Finally, a motif refinement step is applied with the output of the algorithm used as the input of an Expectation Maximization (Lawrence, 1990) algorithm. It guarantees a high probability of finding the solution. The algorithm needs at least 5 or 6 input parameters. Some can be estimated from the input data and the characteristics of the target motifs. In (Buhler, 2001), the details on choosing the parameters values and on the statistical evaluation of the obtained patterns are described. Experiments on synthetic data demonstrated that the algorithm is fast and effective.

The Random Projection algorithm exhibits a complexity that is linear with relation to the length and number of the input sequences. This efficient solution deals well the scalability issues associated with the problem, in contrast to Winnower that in some cases shows a very high time-complexity.

These approaches have two major drawbacks. The first is the randomized nature of the algorithms. These are iterative algorithms that eventually converge to a good solution but no guarantees to find the optimal solution is given. The second is the definition of the extracted patterns, which requires that the l and d parameters are always provided in advance. This is an important limitation since such information is not always available.

Figure 8. Tree structure of the search space.



Approximate Exhaustive Search

The *Weeder* algorithm (Pavesi, 2001) overcomes some of the limitations described in the previous section. Weeder is an “almost” exact algorithm. It is based on the idea introduced by Sagot (Sagot, 1998). The method starts with an exhaustive pattern enumeration (making use of a suffix tree) where ambiguous patterns correspond to a set of paths on the tree. The support of a string s that ends at the node x is given by the number of subtrees rooted at x . Thus, the support of a pattern is obtained by the sum of all the subtrees rooted at the end of its path. The procedure starts with all paths of length d with enough support. Each path corresponds to a valid model. Next, path-extensions are performed, while the number of mismatches has not been reached or the support of the pattern does not drop below the minimum support value. The complexity of this approach is exponential with relation to the number of mismatches, d .

Pavesi et al. introduced the concept of finding patterns where the number of mismatches is proportional to the length of the pattern. The algorithm receives as input a set of sequences and an error ratio $\varepsilon < 1$. Patterns are reported if they have at most $\varepsilon \cdot l$ mismatches. The main restriction is imposed on the location of the mismatches. A valid occurrence of a pattern can have at most $\varepsilon \cdot i$ mismatches in the first i symbols. For example, if the error ratio is set to $\varepsilon = 0.25$ all the paths of length four with more than one mismatch are eliminated. The remaining paths are successively extended. For a pattern of length 16 and the above ε value, 1 mismatch is allowed in the first 4 symbols, 2 in the first 8 symbols, 3 in the first 12 symbols and 4 in the entire pattern. This heuristic introduces a performance improvement when compared with the original Sagot’s proposal. This improvement is achieved by reducing the number of valid occurrences of each pattern, instead of a reduction in the number of patterns.

This algorithm makes two important contributions: (1) it allows, through an exhaustive enumeration, to report patterns longer than those usually obtained by exact methods. (2) The introduction of a number of mismatches proportional to the length of the pattern, through a relative error ratio, does not require the input of the exact length of the patterns nor the number of mismatches. This represents an advantage over exhaustive graph search and random projection search. Since the algorithm is fully based on a heuristic, it results that those patterns which do not have the exact characteristics that are being searched will be missed.

Pattern Enumeration

When searching for longer and ambiguous patterns with greater flexibility, exhaustive search based approaches are not feasible. The pattern enumeration approach is a major alternative. Two types of pattern enumeration are possible: *bottom-up* and *top-down*.

In the bottom-up approach, the key idea is to start with short frequent patterns and keep extending them until the support drops below the minimum support value. When extracting patterns from the closed class we report patterns that its extension leads to support drop. In both cases the process stops when extensions can no longer be performed. This approach makes use of two important properties:

Definition 19 (Monotonic property): All subsequences/substring of a frequent sequence are frequent.

Definition 20 (Anti-monotonic property): All supersequences of an infrequent sequence are infrequent.

The search is then performed in a tree structure like the one shown in Figure 8, which expresses the tree of all possible sequences. In the bottom-up approach the search space is traversed in a Breadth-first (BFS) or Depth-first manner (DFS). In BFS mode, patterns with the same length are extended at the same time, level by level. From Figure 6 the following enumeration is obtained: Level 0: $\{\emptyset\}$, Level 1: $\{a, b\}$, Level 2: $\{aa, ab, ba, bb\}$, Level 3: $\{aaa, aab, aba, abb, baa, bab, bba, bbb\}$, In DFS mode, a sequence pattern is successively extended until it stops fulfilling the imposed conditions. Then, the algorithm backtracks in the search tree and extends a new pattern. The obtained enumeration is in this case: $\{\emptyset, a, aa, aaa\}$, $\{\emptyset, a, aa, aab, aaba\}$, $\{\emptyset, a, aa, aab, aabb\}$, ...

In the top-down approach the algorithm starts with longer sequences. Then, symbol generalizations and length decrease operations are successively applied until the obtained substrings become frequent and can be reported as patterns. The *ToMMS* algorithm (Ester, 2004) is an example of a top-down approach. It reports ambiguous patterns in the presence of substitution sets/concept graphs (introduced later in this chapter).

The main advantage of the pattern enumeration approach is that it allows finding longer and more complex patterns than the exhaustive search. Next, two algorithms based on this approach are described. The Pratt algorithm (Jonassen, 1995), which finds high quality flexible gap patterns by performing pattern extension one symbol at a time. The algorithm Teiresias finds rigid gap patterns by performing multiple symbol extensions.

Single Symbol Enumeration

The *Pratt* algorithm (Jonassen, 1995; Jonassen, 2000) finds rigid and flexible gaps patterns that occur in the majority of the input sequences. This algorithm is an improvement of the method proposed by Neuwald and Green (Neuwald, 1994), that combines a depth first search with the use of a data structure called “blocks”. It allows for a quick check of the occurrences and respective support of a potential pattern. This approach was initially proposed for finding rigid gap patterns with a maximum length, a limited number of concrete events and ambiguous positions. Since the minimum support constraint allows an efficient pruning of the search space, the above methodology was extended by Jonassen et

al. to also extract flexible gap patterns. The Pratt algorithm works in two phases. In the first phase, an enumeration of all the frequent patterns that fulfill the user constraints is obtained. The most significant ones, according to their proposed significance measure, are added to a list of patterns. In a second phase, these patterns are refined and reported to the user.

The basic algorithm for extracting fixed gap patterns works as follows: given an input set of sequences I , all the segments of length k (k -segments) of the sequences in I are enumerated. For each symbol a in Σ and for each position i to k , a set $b_{i,a}$ contains a pointer to all the segments having a in position i . This is calculated for each sequence S in I , denoted B_k^S . The set of all the $b_{i,a}$ for sequences in I composes the blocks data structure, which is used for fast lookup of the segments that match a given pattern. Next, given a minimum support σ , a depth first search is performed. It starts with the empty pattern that matches all the existing segments. A frequent pattern P is extended by a fixed wild-card region and a symbol a from Σ , in the form $P' = P \cdot x(i) \cdot a$. The set of segments of P' can be efficiently calculated intersecting the occurrences of P and the blocks $b_{|P|+i+1,a}$. The patterns are successively extended until its support drops below σ .

Note that when one extension is performed all the possibilities are tried out. That is, all the combinations of all gap lengths ($i \geq 0$ and $i \leq \text{maxGap}$) and all the symbols from Σ . In this case, the components of the pattern (consider the Formula (1)) consist only in concrete events, but ambiguous positions can also be considered. The number of ambiguous positions is defined by the user. For a set A of ambiguous symbols, the respective segment occurrences are included in the block data structure and the recursive depth first search follows as in the case of concrete events.

This method has been extended to extract flexible gap patterns. A flexible gap pattern P consists of a finite set of rigid gap patterns, $L(P)$, that comprises it. The size of this set is given by the product of the flexibilities that compose P . According to Formula (1) and if P has n components then the size corresponds to $\prod_{i=1}^{n-1} (q_i - p_i)$. For instance, if $P = A \cdot x(1,2) \cdot C \cdot x(3,4) \cdot E$, then $L(P) = \{A \cdot x(1) \cdot C \cdot x(3) \cdot E, A \cdot x(1) \cdot C \cdot x(4) \cdot E, A \cdot x(2) \cdot C \cdot x(3) \cdot E, A \cdot x(2) \cdot C \cdot x(4) \cdot E\}$. As introduced above, every pattern $Q \in F(P)$ has its own occurrence list given by the respective block data structure.

Therefore a frequent pattern P is extended into $P' = P \cdot x(i,j) \cdot C$, where C is a component that consist of concrete or ambiguous symbols. Once again all values $i \leq j$, between a minimum and maximum gap lengths are tried. The new pattern P' consists of the set of the patterns Q_t that result from the extension of all the patterns $Q \in F(P)$. The occurrences of a new pattern Q_t is given by $B_{Qt} = B_Q \cup b_{|P|+t+1,C}$. The support of P' is simply the sum of the support of each pattern in $F(P')$.

In order to offer the user the opportunity to restrain the search and to keep the mining process within a reasonable time, Pratt is capable of handling a number of constraints: maximum length of the patterns, maximum number of components, maximum length of a fixed gap, minimum and maximum gap and maximum product of flexibilities.

Two types of refinements can be made in the second step of the algorithm, which will eventually lead to the high scoring patterns. The first consists of extending the length of a pattern P with rigid gaps and ambiguous components and substituting positions in fixed length don't care regions by ambiguous symbols. This approach is fast. But since it is greedy no guarantee of finding all the conserved patterns is given. The second refinement consists of applying specialization operations, where don't care symbols of a pattern P are replaced until it is no longer frequent. It is a much more computationally expensive approach but it ensures completeness.

These two refinements combined with the use of a significance measure to prune out the non high scoring patterns, makes the design of the Pratt algorithm specially suitable for finding only those patterns

that are highly conserved in the set of input sequences. This means that Pratt is not very well suited for extracting patterns whose support is significantly smaller than the number of input sequences. Another critical aspect is that a very particular significance measure is used as a pruning mechanism. This will certainly leave out patterns that could be considered significant according to other criteria.

Multiple Symbol Enumeration

The *Teiresias* algorithm (Rigoutsos, 1998; Floratos, 1999) extracts rigid gap patterns over the alphabet $(\Sigma \cap \{.\})$. The general idea is to find longer patterns by combining shorter ones. First, it extracts all possible short patterns that fulfill a defined density constraint. Next, by combining these initial patterns, an exhaustive search is performed. The density of the pattern defines the minimum number of concrete symbols that the pattern must contain. This limits the number of don't care symbols that occur in any stretch of the pattern.

Given the integers L and W , with $L \leq W$, a pattern P is called a $\langle L, W \rangle$ pattern, if it is a string over $(\Sigma \cap \{.\})$. It starts and ends with a symbol from Σ . Every subpattern of P , which contains exactly L symbols from Σ , is no longer than W . For example, $AF \dots CH \dots E$ is a valid $\langle 3, 5 \rangle$ pattern, while $AF \cdot C \cdot H \dots E$ is not, since the subpattern $C \cdot H \dots E$ is 6 symbols long. For a pattern P and a database I of sequences, the offset list ol of P is a list of pairs that contains the sequence-identifier and the offset value of the pattern in that list. For the input set $I = \{\text{LFAADCIFFEDTR, LKLADCHFFELSDR, AFAGCADVHFF}\}$ and the pattern $P = AD \dots FF$, the offset list is $ol(P) = \{(1,4), (2,4), (3,6)\}$.

Definition 21 (Specific Patterns): A pattern P' is *more specific* than P , if P' specializes one or more symbols of P and/or extends P by one or more symbols.

For example $AB \cdot CD$ is more specific than $AB \dots D$.

Definition 22 (Teiresias Maximal Patterns): A pattern P is defined to be *maximal* with relation to the input sequences I if there is no other pattern Q , more specific than P having the same set of occurrences in I .

This definition states that all the patterns that cannot be extended without dropping its support, i.e. closed patterns, are consider *potentially maximal*. From these, the most specific ones are considered effectively maximal. For example, if $AB \cdot D$ and $AB \cdot DE$ have the same support, the latter is maximal since it is more specific than the former.

Teiresias algorithm works in two phases: *scanning* and *convolution*. In the scanning phase, the database sequences are scanned and all the elementary patterns with at least support σ are extracted. An elementary pattern is a $\langle L, W \rangle$ pattern that contains *exactly* L symbols from Σ . The scanning is done through an exhaustive search as in the Pratt algorithm. Additionally, the offset list for each of these patterns is obtained. In this phase, the algorithm breaks up the sequences into smaller pieces that are the basic blocks for the next phase. In the convolution phase, the elementary patterns of the previous phase are successively combined into larger patterns until all the maximal patterns have been generated. The idea is that an original pattern P can be reconstructed from a pair of intermediate patterns A and B such that the suffix of A is equal to the prefix of B .

The reconstruction is done through a binary operation called *convolution*. This operation makes use of two auxiliary functions *Prefix* and *Suffix*, which respectively output the prefix and suffix subpatterns, like for example: $\text{Prefix}(FASTS) = 'FA'$ and $\text{Suffix}(FASTS) = 'STS'$.

Definition 23 (Convolution): The convolution operation is defined as follows: let P, Q be arbitrary patterns with at least L symbols each. The patterns P and Q are combinable if the prefix of P is equal to the suffix of Q, both containing exactly L-1 symbols. P and Q are then merged so that the L-1 concrete symbols overlap. The convolution operation is denoted by $P \circ Q$.

For instance, $AB . CD . E \circ DFE . G = \emptyset$, because $D . E \neq DF . .$. However, $AB . CD . E \circ D . E . G = AB . CD . E . G$. If the offset list of these two patterns are $\{(1,1),(2,6),(4,7)\}$ and $\{(1,4),(3,5),(4,10)\}$, then the offset list of the new pattern is $\{(1,1),(4,7)\}$. Figure 9 depicts an example, where a longer pattern is obtained from the convolution of several smaller patterns.

Some of the patterns generated in the convolution phase are non-maximal or may occur repeatedly. Thus, all the frequent patterns are stored in a list, and each new pattern is compared with the existing patterns in this list. This allows the elimination of less specific patterns, ensuring that any maximal pattern P is always reported before any non-maximal pattern subsumed by P. The order in which patterns are generated is achieved through the introduction of two partial orders based on the prefix and suffix ordering of the set of the patterns.

Teiresias is an exact algorithm, guaranteed to find all the maximal $\langle L, W \rangle$ patterns with at least support equals to σ . Experimental evaluation demonstrates that this algorithm is linear with the number of reported patterns (Rigoutsos, 1998; Floratos, 1999). On average the algorithm runs fast, but it can have an exponential time complexity in the worst case. The first version of Teiresias only allowed patterns over the $(\Sigma \cap \cdot)^*$ language. A later version (Rigoutsos, 2000) handles substitution sets and therefore extracts patterns over $(\Sigma \cap \cdot \cap R)^*$. The major drawback of the algorithm is the limitation to $\langle L, W \rangle$ patterns, which in many cases may not be flexible enough to express less well conserved regions.

Top-Down Enumeration

The ToMMS algorithm (Ester, 2004) proposes a top-down approach to find longer ambiguous patterns. It starts the search with longer infrequent concrete patterns. Then, two operations are successively applied until these patterns become frequent. These operations are: *length-reduction*, where symbols from the extremities are deleted and *generalizations*, where symbols of the patterns are substituted by other symbols in superior level of the concept graph. A *concept graph* is a graph that extends the concept of substitution sets for multiple hierarchical levels. In this graph different levels correspond to different degrees of “*generality*”. Lower levels, correspond to the symbols of the alphabet and are more specific. Upper levels contain symbols that represent concepts and are more general.

Note that in the top-down traversal of ToMMS, only a small subset of generalizations will be applied to a pattern. These generalizations should actually increase the support. The general idea of the algorithm can be described as follows: given an input set I , a minimum support σ , a maximum length max_len and

Figure 9. Example of a pattern reconstruction by the convolution operation. Elementary patterns are signaled in grey.

$$\begin{array}{c}
 \text{F.AS} \\
 \oplus \rightarrow \text{F.AST} \\
 \text{AST} \quad \oplus \rightarrow \text{F.ASTS} \\
 \text{STS}
 \end{array}$$

a concept graph T , start with the set of all infrequent concrete patterns of length max-len. Next, while the set of infrequent concrete sequences is not empty:

- apply a length-reduction to generate pattern substrings;
- determine all pairwise matches;
- perform minimal generalizations by using the concept graph T ;

If the newly generated pattern is frequent, it is reported and the generation of its substrings is stopped; otherwise apply this procedure to all its substrings. Through the pairwise matching of the substrings (step 2) only sequences that generalize two infrequent sequences are generated.

The main advantages of this approach are the fact that longer patterns are found and the more specific patterns are traversed first than the less specific ones. The major drawback of the approach is the fact that it only reports ambiguous patterns. Another drawback is the need to estimate the maximum length of the frequent patterns. The authors proposed a solution where a simplified version of the algorithm is applied iteratively until the correct length is found, which may only be feasible for some cases.

Summary of Algorithms Features

Table 1 and Table 2 present the summary of the characteristics of the algorithms proposed for biological sequence mining.

CONCLUSION

In this chapter we have reviewed the process of genetic sequence pattern mining. We have provided a general and abstract architecture for a sequence mining algorithm, where each of its blocks was discussed individually. Depending on the type of data, the format of the input sequences, the type of patterns to be extracted and the background knowledge provided by the user, different algorithmic approaches can be used. We have surveyed methods and algorithms designed to mine DNA and protein sequence databases.

Table 1. Algorithm and characteristics of outputted patterns.

Algorithm	Length	Type	Class	Best Pattern
eMotif	Fixed	Rigid	All	no
Verbumculus	Variable	Concrete	All	no
Winnower	Fixed	Ambiguous	All	yes
Random Projection	Fixed	Ambiguous	All	yes
Weeder	Variable	Ambiguous	All	no
Pratt	Variable	Rigid/Flexible	All	yes
Teiresias	Variable	Rigid	Closed/Maximal	no
ToMMS	Variable	Ambiguous	Maximal	no

Table 2. Algorithm and characteristics of the mining process, respectively: direction of the traversal, traversal mode and application of heuristics.

Algorithm	Traverse Direction	Traverse Mode	Heuristics
eMotif	NA	NA	no
Verbumculus	Bottom-Up	DepthFirst	no
Winnower	NA	NA	yes
Random Projection	NA	NA	yes
Weeder	Bottom-Up	DepthFirst	yes
Pratt	Bottom-Up	DepthFirst	yes
Tieresias	Bottom-Up	NA	no
ToMMS	Top-Down	NA	yes

Genetic sequence pattern mining is a very active field of research that has attracted the attention of the research community for more than twenty years. Complemented by the ever growing knowledge of the molecular biology mechanisms and the development of new sequencing technologies this will certainly keep being one of the most important topics of bioinformatics research in the near future.

REFERENCES

- Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., & Walter, P. (2002). *Molecular biology of the cell* (4th ed.). London: Garland Science.
- Apostolico, A., Comin, M., & Parida, L. (2005). Conservative extraction of over-represented extensible motifs. *Bioinformatics (Oxford, England)*, 21(Suppl 1), i9–i18. doi:10.1093/bioinformatics/bti1051
- Apostolico, A., Gong, F., & Lonardi, S. (2004). Verbumculus and the discovery of unusual words. *Journal of Computer Science and Technology*, 19(1), 22–41. doi:10.1007/BF02944783
- Baldi, P., Brunak, S., Chauvin, Y., Andersen, C., & Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: An overview. *Bioinformatics*, 16(5), 412–422. doi:10.1093/bioinformatics/16.5.412
- Brazma, A., Jonassen, I., Eidhammer, I., & Gilbert, D. (1998a). Approaches to the automatic discovery of patterns in biosequences. *Journal of Computational Biology*, 5(2), 279–305. doi:10.1089/cmb.1998.5.279
- Brazma, A., Jonassen, I., Vilo, J., & Ukkonen, E. (1998b). Pattern discovery in biosequences. In *Proceedings of the 4th International Colloquium on Grammatical Inference* (LNAI 1433, pp. 255–270). Berlin, Germany: Springer.
- Brazma, A., Parkinson, H., Schlitt, T., & Shojatalab, M. (2001). A quick introduction to elements of biology - cells, molecules, genes, functional genomics microarrays. Retrieved October 2001, from <http://www.ebi.ac.uk/2can/tutorials/index.html>

- Buhler, J., & Tompa, M. (2001). Finding Motifs using random projections. In *Proceedings of the 5th International Conference on Computational Molecular Biology* (pp. 69-76).
- Califano, A. (2000). SPLASH: Structural pattern localization analysis by sequential histograms. *Bioinformatics (Oxford, England)*, 16(4), 341–357. doi:10.1093/bioinformatics/16.4.341
- Cooper, N. (1994). *The Human genome project, deciphering the blueprint of heredity*. New York: University Science Books.
- Das, M., & Dai, H. (2007). A survey of DNA motif finding algorithms. *BMC Bioinformatics*, 8(Suppl 7), S21. doi:10.1186/1471-2105-8-S7-S21
- Durbin, R., Eddy, S., Krogh, A., & Mitchison, G. (1998). *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge, UK: Cambridge University Press.
- Ester, M., & Zhang, X. (2004). A top-down method for mining most specific frequent patterns in biological sequence data. In *Proceedings of the 4th SIAM International Conference on Data Mining*.
- Ferreira, P., & Azevedo, P. (2007). Evaluating deterministic motif significance measures in protein databases. *Algorithms for Molecular Biology; AMB*, 2(16).
- Floratos, A. (1999). *Pattern discovery in biology: Theory and applications*. Unpublished doctoral dissertation, University of New York, USA.
- Gusfield, D. (1999). *Algorithms on strings, trees and sequences: Computer science d computational biology*. Cambridge, UK: Cambridge University Press.
- Han, J., & Kamber, M. (2001). *Data mining concepts and techniques*. San Francisco: Morgan Kaufmann.
- Henikoff, S., & Henikoff, J. (1993). Performance evaluation of amino acid substitution matrices. *Proteins*, 17(1), 49–61. doi:10.1002/prot.340170108
- Huang, J., & Douglas, B. (2001). The eMOTIF database. *Nucleic Acids Research*, 29(1), 202–204. doi:10.1093/nar/29.1.202
- Hulo, N., Bairoch, A., Bulliard, V., Cerutti, L., De Castro, E., & Langendijk-Genevaux, P. (2006). The PROSITE database. *Nucleic Acids Research*, 34, D227–D230. doi:10.1093/nar/gkj063
- Hunter, L. (1993). Molecular biology for computer scientists. In L. Hunter (Ed.), *Artificial intelligence and molecular biology* (pp. 1-46). Menlo Park, CA: AAAI Press.
- John, R., & Holm, J. R. (1995). *Elements of general, organic and biological chemistry* (9th ed.). New York: John Wiley and Sons.
- Jonassen, I. (2000). Methods for discovering conserved patterns in protein sequences and structures. In D. Higgins, & W. Taylor (Eds.), *Bioinformatics: Sequence, structure and databanks. A practical approach*. Oxford, UK: Oxford University Press.
- Jonassen, I., Collins, J., & Higgins, D. (1995). Finding flexible patterns in unaligned protein sequences. *Protein Science*, 4(8), 1587–1595. doi:10.1002/pro.5560040817

- Koonin, E., & Galperin, M. (2003). *Sequence-evolution-function: Computational approaches in comparative genomics*. Amsterdam: Kluwer Academic Publishers.
- Krogh, A. (1998). An introduction to Hidden Markov Models for biological sequences. In *Computational methods in molecular biology* (pp. 45-63). St. Louis, MO: Elsevier.
- Lawrence, C., & Reilly, A. (1990). An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned byopolymer sequences. *Proteins*, 7(1), 41–51. doi:10.1002/prot.340070105
- Lesk, A. M. (2002). *Introduction to bioinformatics*. Oxford, UK: Oxford University Press.
- Li, N., & Tompa, M. (2006). Analysis of computational approaches for motif discovery. *Algorithms for Molecular Biology; AMB*, 1(8). doi:10.1007/978-1-59745-000-3
- Lonardi, S. (2002). Pattern discovery in biosequences - tutorial. In *Proceedings of the 10th International Conference on Intelligent Systems for Molecular Biology*.
- McCreight, E. (1976). A space economical space tree construction algorithm. *Journal of the ACM*, 23(2), 262–272. doi:10.1145/321941.321946
- National Institutes of Health. (2007). *The human genome project*. Retrieved from <http://www.genome.gov/>
- Neuwald, A., & Green, P. (1994). Detecting patterns in protein sequences. *Journal of Molecular Biology*, 239, 698–712. doi:10.1006/jmbi.1994.1407
- Pavesi, G., Mauri, G., & Pesole, G. (2001). An algorithm for finding signals of unknown length in DNA sequences. *Bioinformatics (Oxford, England)*, 17, S207–S214.
- Pevzner, P., & Sze, S. (2000). Combinatorial approaches to finding subtle signals in DNA sequences. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology* (pp. 269-278). Menlo Park, CA: AAAI Press.
- Rigoutsos, I., & Floratos, A. (1998). Combinatorial pattern discovery in biological sequences: The Teire-sias algorithm. *Bioinformatics (Oxford, England)*, 14(1), 55–67. doi:10.1093/bioinformatics/14.1.55
- Rigoutsos, I., Floratos, A., Parida, L., Gao, Y., & Platt, D. (2000). The emergence of pattern discovery techniques in computational biology. *Metabolic Engineering*, 2(3), 159–177. doi:10.1006/mben.2000.0151
- Sagot, M.-F. (1998). Spelling approximate repeated or common motifs using a suffix tree. In *Theoretical informatics* (LNCS 1380, pp. 111-127). Berlin, Germany: Springer.
- Sandve, G., & Drabløs, F. (2006). A survey of motif discovery methods in an integrated framework. *Biology Direct*, 1(11).
- Tompa, M. (2005). Assessing computational tools for the discovery of transcription factor binding sites. *Nature Biotechnology*, 23(1), 137–144. doi:10.1038/nbt1053

Ukkonen, E., Vilo, J., Brazma, A., & Jonassen, I. (1996). Discovering patterns and subfamilies in biosequences. In *Proceedings of the 4th International Conference on Intelligent Systems for Molecular Biology* (pp. 34-43). Menlo Park, CA: AAAI Press.

Wang, L., & Jiang, T. (1994). On the complexity of multiple sequence alignment. *Journal of Computational Biology*, 1(4), 337–348. doi:10.1089/cmb.1994.1.337

Weiner, P. (1973). Linear pattern matching algorithm. In *Proceedings of the 14th IEEE Symposium on Switching and Automata Theory* (pp. 1-11).

Witten, I., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques*. San Francisco: Morgan Kaufmann.

Wu, T., & Brutlag, D. (1995). Identification of protein motifs using conserved amino acid properties and partitioning techniques. In *Proceedings of the 3rd International Conference on Intelligent Systems for Molecular Biology* (pp. 402-410).

KEY TERMS AND DEFINITIONS

Sequence Motif: pattern that captures a sequence region that is conserved in multiple sequences and has an expected biological function.

Pattern Mining: a data mining technique to discover patterns in data. Genetic Sequences sequences that encode genetic information. Typically correspond to DNA, RNA or proteins sequences.

Motif Discovery: methods to discover sequence motifs in genetic sequence data.

Search Algorithms: strategies applied in motif discovery algorithms to discover patterns in data.

Pattern Evaluation: procedure based on a function that provides a score to a sequence pattern expressing its statistical significance.

Sequence Alignment: arrangement of genetic sequences in order to emphasize the regions conserved across the different sequences.

Chapter 14

Machine Learning in Natural Language Processing

Marina Sokolova
CHEO Research Institute, Canada

Stan Szpakowicz
University of Ottawa, Canada and Polish Academy of Sciences, Poland

ABSTRACT

This chapter presents applications of machine learning techniques to traditional problems in natural language processing, including part-of-speech tagging, entity recognition and word-sense disambiguation. People usually solve such problems without difficulty or at least do a very good job. Linguistics may suggest labour-intensive ways of manually constructing rule-based systems. It is, however, the easy availability of large collections of texts that has made machine learning a method of choice for processing volumes of data well above the human capacity. One of the main purposes of text processing is all manner of information extraction and knowledge extraction from such large text. Machine learning methods discussed in this chapter have stimulated wide-ranging research in natural language processing and helped build applications with serious deployment potential.

NATURAL LANGUAGE PROCESSING AS A CHALLENGE FOR MACHINE LEARNING

The science and technology of the use and properties of human languages is generally known as Natural Language Processing (NLP). The name emphasizes, on the one hand, the natural origins of human spoken languages and, on the other, their role in computing. The discipline goes by several other mostly self-explanatory names, each bringing into focus a different key aspect: Computational Linguistics, Natural Language Understanding, Language Technology, Language Engineering, Text Processing, Text Analysis, etc. Disciplines that deal with speech, for example Speech Recognition and Speech Generation, may or may not be included in Natural Language Processing. Major classes of applications of theories, tools

Table 1. Algorithms and corresponding learning problems discussed in some detail in this chapter.

Algorithm	Problem
Clustering by Committee	Learning concepts from text
Decision Trees	Classification of texts with features that are redundant (relevant but nor discriminative enough)
Deep Neural Networks	Combined semantic and probabilistic language modeling; multi-task feature learning
Finite-state automata	Stemming
Support Vector Machine	Classification of texts in high-dimensional feature spaces
Transformation-based learning	Part-of-speech tagging

and techniques developed in NLP include Machine Translation, Text Summarization, Text Mining and Information Extraction (IE). We refer the reader to (Mitkov, 2003) for an up-to-date description of NLP tasks and techniques. Feldman and Sanger (2007) discuss IE, visualization methods and probabilistic modelling, which often complement Machine Learning (ML) in language analysis. Sebastiani (2002) and Lopez (2008) present comprehensive overviews of ML applications to text categorization and translation, respectively.

Vocabulary, grammar, style – a few of many salient language dimensions – depend upon central dichotomies in natural language: spoken/written, edited/spontaneous, objective/subjective, formal/informal, and so on (Biber, 1988; Crystal, 2006). The nearly inexhaustible richness of language characteristics is “a challenge to those minds which seek ordered simplicity in the world, and at the same time a collectors’ paradise” (Firth, 1936). The fast-increasing volume of readily available digital texts makes natural language not only a fertile ML research area (Daelemans, 2006), but also one of the most important data formats for ML applications (Shawe-Taylor and Christianini, 2004). The performance of ML algorithms is routinely compared on document topic classification, word-sense disambiguation and opinion analysis, to name just a few common NLP tasks.

This chapter presents ML applications to fundamental language-processing and linguistic problems: identify a word’s part-of-speech or its meaning, find relations between words, and so on. Such problems, once solved, are a necessary component of many a higher-level language processing task, including text summarization, question answering and machine translation.

The next section of the chapter positions ML applications among technological tools for NLP. Next, we look at NLP from the ML perspective, focusing on such issues as the importance of annotated data, the characteristics of text data sets and the evaluation measures. We then discuss the nature of NLP problems solved by ML methods, which we propose to see as *nested learning* – based on the results of other learning. We show how Named-Entity Recognition, part-of-speech tagging and parsing contribute to learning in more advanced problems NLP problems. We look in some detail at ML applications to Word-Sense Disambiguation (WSD), one of the core NLP problems. Table 1 sums those algorithms and problems to which we have paid particular attention in this chapter. We discussed more algorithms and problems, but with fewer details.

Further in this handbook, the chapter “Machine Learning Applications in Mega-Text Processing” discusses in detail a few NLP tasks that emerged when text storage and exchange in electronic form became the norm.

THE POSITION OF MACHINE LEARNING AMONG OTHER NATURAL LANGUAGE PROCESSING TECHNOLOGIES

Natural Language Processing is an area of research and development which deals with language and language data, spoken (digitized and transcribed) and written. Ideally, NLP methods would process manageable amounts of text nearly as well as people do, and would effortlessly analyze large volumes of text. For example, automated part-of-speech (POS) tagging should be close to the nearly flawless performance of a qualified human tagger. Tagging software should label *housing* as an adjective in *the housing project* and as a gerund in *housing the project*. In corpora of millions of words, easily available now, a tagger must work well across contexts and contents. POS taggers do approach such performance, but more advanced language tasks perform far from ideally. The illustrative examples are Machine Translation, where an NLP system should automatically *and accurately* map expressions in a source language onto a target language, and various forms of text understanding, where a system should be able summarize text, determine the level of novelty, give coherent answers to questions about the text, and so on, and so forth.

Over the years, there have been three broad styles of work with NLP tasks: symbolic processing, statistical analyses and Machine Learning. Symbolic processing – the leading paradigm from the inception of NLP in the mid-1950s till around the 1980s – is linguistically the most accurate but very labour-intensive. It requires substantial effort of suitably trained personnel. Hand-crafted morphological analyzers, rule-based syntactic analyzers, domain-specific knowledge bases, rule-based translation are all examples of the symbolic approach (Stenning et al, 2006). Symbolic methods may provide gold standards for part-of-speech tagging, word-sense disambiguation, sentence parsing, summarization, translation and so on. They are, however, ill-equipped to handle the amounts of text supplied by the Internet and other electronic sources and data reservoirs.

Statistical NLP (SNLP) can be understood to build on the distributional hypothesis (Harris, 1968); Samuelsson (2004) restates this position. On the other hand, SNLP is also seen as a combination of all quantitative approaches to automated language processing, including probabilistic modeling, information theory and linear algebra (Manning and Schütze, 2003). Both views assume that we can “judge a word by the company it keeps” (Firth, 1957). Much about language can be discovered from large amount of data by counting words, word sequences and their contexts (Kilgarriff, 2001). Word distribution in a large corpus is approximately described by the Zipf law (Zipf, 1935):

$$f \cdot r = k \quad (1)$$

f is the frequency of a word in the corpus, r is its rank (the position on a word list ordered by frequency), and k is a constant. Statistical language models, statistical parsers, statistical machine translation are now a fact of life in the NLP community. The statistical methods all work the better the more data is available. There must also be properly organized seed information (such as alignment in parallel multilingual corpora used to train machine translation systems) whose development requires manual work.

Starting in the 1980s, Machine Learning has become a necessity in NLP and is now an essential element of the researcher’s toolkit. ML has replaced symbolic analysis as the paradigm of choice for work with language data. Let us clarify that we understand a computational system to be an ML system if its performance improves with experience (Mitchell, 1997). Such systems may represent knowledge in a symbolic, declarative form (that is what Decision Trees and Decision Lists do), in a numeric format

(optimization-based SVM and probability-based Naïve Bayes), or in model-based ways (neural networks, Hidden Markov Models (HMM)). The ability to acquire new knowledge distinguishes ML methods from symbolic and statistical methods of working with language. For a general overview and discussion of modern language technology we refer the reader to (Jurafsky and Martin, 2008), a systematic survey of speech and language processing, which covers all three styles. The Internet site of the Association for Computational Linguistics¹ is a reference equally excellent for the history of NLP and for the latest advances.

The influence of ML has been tangible in the field of NLP. ML draws attention and gives priority to tasks which fit the default classification-based supervised learning framework and have annotated data readily available (Daelemans, 2006).² Such tasks include text classification by topic and part-of-speech. It has become an important NLP concern to formulate problems in a way appropriate to ML. A well-stated ML problem requires its input and output to be quantified or categorized. A distinct property of input text data, if compared with any numerical data, is a variety of its possible quantitative representations. To decide what properties of the input text will be quantified is an important step in building a proper ML application. The same text can be quantified by its linguistic (e.g., syntactic, semantic or pragmatic) features, text statistics, and so on. Our list does not include phonetic and morphological language characteristics mostly used in applications related to speech.

The categorization of the output is easier when it is based on objective factors, as it is in document classification by topic. When the output of human cognitive activities is inherently subjective, NLP problems tend to become complex. It may be necessary to partition such problems into subproblems feasible for ML applications. Among the studies of subjective language that use ML techniques, *opinion mining* is a well-developed task. Opinion mining can be applied to professionally written articles (Wiebe and Riloff, 2005), customer-written reviews (Hu and Liu, 2004) or political debates (Thomas et al, 2006). Its elevated position among ML applications may be explained by the deployment of a rich ML apparatus to the solving of a series of well-defined subproblems, among them learning the opinion value of separate words or single utterances, relations between representations of positive and negative opinions and the importance of neutral opinions.

To choose the appropriate tools requires care. That is partially because not all language tasks must be solved by complex means. Some are better served by rules, sometimes as simple as those that drive finite-state automata. *Stemming* exemplifies such tasks. A stemmer removes selected inflectional endings following rather few and rather simple rules. This reduces the word to a likely stem; for example, *holds* and *holding* become *hold*. Porter's rule-based algorithm³ is a popular stemmer, originally designed for English (Porter, 1980). It has five rules, each applied in one to three steps. There is no learning, no performance improvement.⁴ Stemming is a popular pre-processing technique that can substantially decrease the number of word features, but empirical evidence suggests that its use may not change learning results significantly. Regardless of the presence of stemming, Naïve Bayes, SVM, k Nearest Neighbour and the tf·idf measure⁵ applied to two different email corpora (represented by five different feature sets) classified spam email equally well (Lai and Tsai, 2004). Note that stemming differs from the much more precise process of lemmatization, which accounts for irregular forms and derivational affixes; for example, *held* and *holding* have the same lemma *hold*.

(Jurafsky and Martin, 2008; Manning and Schütze, 2003) are the comprehensive reference books for the reader interested in these fundamental NLP tasks and their algorithmic solutions, including methods based on IE and SNLP.

A MACHINE LEARNING PERSPECTIVE ON NATURAL LANGUAGE PROCESSING

When we discuss applications of ML to NLP problems, we want to remember that manually annotated (labelled) training sets are very expensive to build. In many applications researchers use either automatically annotated sets or self-annotated sets (marked by data contributors). Either kind tends to have labels that are not as reliable as those produced manually. On the other hand, such less reliable sets usually supply a substantially higher number of data entries; compare 8000 self-annotated entries of consumer-written product reviews (Dredze et al, 2007) with 314 manually annotated entries of the same data (Hu and Liu, 2004). For some NLP tasks, ML methods can give adequate results only when used on large amounts of data. One of the most striking examples is Statistical Machine Translation. To output reliable results for a new pair of languages, the ML methods would require more than a *million* words of parallel data (mutual translations, typically aligned by sentence) and two monolingual data sets of more than a *billion* words each (Och, 2005). NLP problems well served by ML methods hold the middle ground: they do not need a mass of annotated data, but they can be solved without labour-intensive hand-crafting of knowledge bases.

Perhaps the favourite ML applications in the sphere of language are those that benefit from data generalization. Still, ML methods encounter challenges that come from data complexity, unbounded even within the scope of written texts. For a text, the language dimensions bring in their specific features (such as the factual vocabulary in objective texts, complex co-reference resolution in interactive communication, longer sentences of elaborative texts). That makes the language data a unique and complex organism. As a result of this uniqueness, the no-free-lunch theorem – “any two algorithms are equivalent when their performance is averaged across all possible problems” (Wolpert and Macready, 2005) – is especially true when one applies ML algorithms to texts.

In some ways, ML borrows from IE and Information Retrieval which were historically a prelude to ML in automated text analysis. Evaluation metrics for ML algorithms are an example of such borrowing. The metrics commonly used in NLP (Precision, Recall, F-score) have their origin in Information Retrieval (Rijsbergen, 1979). Calculated for classifiers produced by an algorithm, they build on the numbers of correctly classified positive examples TP , incorrectly classified positive examples FP , and incorrectly classified negative examples FN . Precision (P) measures how many of the classifier’s verdicts have been correct. Recall (R) measures how few correct verdicts the classifier has missed.

$$P = \frac{TP}{TP + FP} \quad (2)$$

Precision and Recall can be calculated by *micro-averaging* or *macro-averaging* (Sebastiani, 2002). Micro-averaged Precision and Recall are calculated by summing up true and false positives for each class i :

$$P_{\text{micro}} = \frac{\sum_{i=1}^m TP_i}{\sum_{i=1}^m (TP_i + FP_i)} \quad (3)$$

m is the number of classes. Macro-averaged Precision (Recall) is the averaged sum of Precision (Recall) calculated for each class:

$$P_{macro} = \frac{\sum_{i=1}^m P_i}{m} \quad (4)$$

P_i and R_i are calculated for each class. Micro-averaging takes into account frequencies of class labels and favours correct classification of classes with many positive examples. Macro-averaging puts equal weights on all classes regardless of how few positive examples they have. A commonly applied combined measure of the quality of text classification is F-score, a weighted mean of Precision and Recall:

$$F = \frac{(\beta^2 + 1)P}{(\beta^2 + 1)P + R} \quad (5)$$

It is customary to apply equal weighting, $\beta = 1$. If $\beta > 1$, F-score favours Precision over Recall. $\beta < 1$ weights Recall over Precision. Micro-averaged F-score is calculated from P_{micro} and R_{micro} , macro-averaged F-score – from P_{macro} and R_{macro} .

The omission of correctly classified negative examples TN is a reminiscence of the former dominance of document classification by topic. In applications which treat positive and negative examples equally, the standard Accuracy (A) is used for performance assessment:

$$A = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

Performance in ML is usually evaluated by applying algorithms to one or several data sets and calculating one or more of the measures listed above. There are benchmark data sets (e.g., 20 Newsgroups⁶, Reuters-21578⁷) for topic classification of large document collection, which assist in direct comparison of algorithm performance. There are no such benchmarks for classification other than by topic; refer to the chapter “Machine Learning Applications in Mega-Text Processing” for more information. This absence considerably complicates direct comparison. The applicability of performance measures to subfields of text classification is covered in (Sokolova and Lapalme, 2007).

Across the spectrum of NLP problems, Support Vector Machines (SVM), Decision Trees (DT), k-Nearest Neighbour (kNN) and naïve Bayes (NB) are applied most often. This may be a consequence of the expected performance results, supported by easily available algorithm implementations. An empirical comparison of SVM, DT, kNN and NB showed a correspondence between the characteristics of text data and the algorithm performance (Colas and Brazdil, 2006; Gabrilovich and Markovitch, 2004; Joachims, 2002). NB, a technique that is fast and easy to implement, has the edge when a data set has a small number of entries (Colas and Brazdil, 2006). SVM is a popular technique in text classification. It usually has high accuracy and good F-score in high-dimensional features spaces, where texts are represented by many features relevant to classification (Joachims, 2002). The support vector algorithm, however, does not always work well. For example, texts can contain redundant features: relevant to classification but without additional discriminative power. Such features are common in text categori-

zation data sets built from hierarchical directories of documents. Gabrilovich and Markovitch (2004) show that the C4.5 implementation of DTs outperformed SVMs on data with many redundant features: 80.0% to 76.9% in classification accuracy. With redundant features deleted, SVM improved its accuracy to 85.3%, C4.5 - to 84.3%.

Other algorithms – among them, Structural Correspondence Learning (Dredze et al, 2007), Fuzzy Clustering (Witte and Bergler, 2007), and Conditional Random Fields (Choi et al, 2005) – give highly accurate results on NLP tasks and should become tools of choice for NLP researchers and practitioners. Some learning algorithms are language-oriented by design, for example, Latent Semantic Analysis (LSA). LSA considers co-occurrences of a word with neighbouring words within a given context of a sentence, paragraph or document (Landauer and Dumais, 1997).

Selection of text representation features can make a difference between successful and unsuccessful ML application. Features can be statistically selected after mapping text into numerical format such as the frequency of occurrence of a word in text. Section “Empirical comparison of algorithms in the learning of opinions” in the chapter “Machine Learning Applications in Mega-Text Processing” briefly describes four statistical selection methods. For more detailed description and formulae, a reader can refer to (Manning and Schutze, 2003). Features can be constructed based on semantic or syntactic analysis, for example part-of-speech or word dependence (Jurafsky and Martin, 2008).

Later in this chapter, on an example of learning word senses, Tables 6 and 7 illustrate how different syntactic features cause change in learning results even if the same algorithm is used. Features need not be static: different text fragments can be represented by different features (Blatak et al, 2004). Within a given task, algorithm performance may vary by time-dependent feature selection, as was shown by classifying success and failure of electronic business negotiations from text messages exchanged by negotiators (Sokolova et al, 2008).

Learning from language data is well characterized by the wide use of external information sources such as dictionaries, ontologies or repositories of labelled examples (Agirre and Martinez, 2007; Daelemans, 2006). Let us list three typical applications.

1. *Text classification.* To classify texts, algorithms access ontologies of relevant terms, or dictionaries, or lists of key term lists. Examples include the Medical Subject Headings (MeSH)⁸, the World Gazetteer⁹ and the communicable disease ontology¹⁰. It has been shown that adding the MeSH terms to various semantic and bag-of-word feature sets improves NB’s accuracy of biomedical text topic classification and clustering (Lee et al, 2006).
2. *Syntactic analysis.* The learning of connections between words in phrases or other syntactically motivated groupings relies on treebanks – texts annotated with syntactic structures of sentences. The Penn Treebank (Marcus et al, 1993) consists of several corpora – small by today’s standards – presented as “skeletal parses showing rough syntactic and semantic information”¹¹. Treebanks have been built for other languages, e.g., the Prague Dependency Treebank for Czech¹², and for specific domains, e.g., the GENIA Treebank for biomedical texts¹³. The syntactic information can be used as features in a learning algorithm. In identifying protein-protein interaction, SVM enhanced with syntactic/semantic tree kernels improved the performance over a strong baseline (bag of words). Precision, Recall and F-score rose from 48.2%, 54.9% and 51.1% to 54.9%, 65.6% and 59.5% (Miyao et al, 2008). The paired t-test shows that the differences are statistically significant, p-value = 0.00748 (personal communication).

Table 2. The Penn Treebank part-of-speech Tags (excluding punctuation)

Tag	Part-of-speech	Tag	Part-of-Speech	Tag	Part-of-Speech
CC	Coordinating Conjunction	NNS	Noun, plural	TO	“to”
CD	Cardinal number	NNP	Proper noun, singular	UH	Interjection
DT	Determiner	NNPS	Proper noun, plural	VB	Verb, base form
EX	Existential “there”	PDT	Predeterminer	VBD	Verb, past tense
FW	Foreign word	POS	Possessive ending	VBG	Verb, gerund or present participle
IN	Preposition or subordinating conjunction	PP	Personal pronoun	VBN	Verb, past participle
JJ	Adjective	PP\$	Possessive pronoun	VBP	Verb, non3rd pers. singular present
JJR	Adjective, comparative	RB	Adverb	VBZ	Verb, 3rd person singular present
JJS	Adjective, superlative	RBR	Adverb, comparative	WDT	Wh-determiner
LS	List item marker	RBS	Adverb, superlative	WP	Wh-pronoun
MD	Modal	RP	Particle	WP\$	Possessive wh-pronoun
NN	Noun, singular or mass	SYM	Symbol	WRB	Wh-adverb

3. *Word categorization.* To learn word categories, for example parts of speech, algorithms are trained on annotated corpora. The British National Corpus (Leech et al, 1994)¹⁴, the Brown Corpus (Kucera et al, 1967)¹⁵, the Wall Street Journal (Marcus et al, 1993)¹⁶ - benchmark annotated texts in British or American English - are often used to train learning algorithms. In languages where word boundaries are not obvious or where inflection is much richer than in English, corpora may have a very large set of tags. For example, detailed POS information along with name entities and noun phrase structures help resolve word boundaries in Thai, where words are written without delimiters (Varasai et al, 2008).

Note that POS categories used in NLP are much finer-grained than the basic noun, verb, adjective, adverb, article, preposition categories. Strongly inflected languages have tag sets of hundreds or more tags. “The Czech PDT corpus provides a much richer set of POS tags, with over 3000 possible tags defined by the tagging system and over 1000 tags actually found in the corpus” (Collins et al., 1999).

The British National Corpus uses 57 word class tags (34 of them for verbs), 4 punctuation tags, and 30 tags marking common ambiguities, e.g., between general or positive adjective and singular common noun. Many applications opt for a smaller 36-tag list used in the Penn Treebank data. Table 2 lists these tags in their alphabetical order.

NESTED LEARNING

An NLP task can be an example of what we will refer to as *nested learning*¹⁷: the application of ML methods relies on the results of other learning. Consider translation and re-ordering of words, essential tasks in Machine Translation. They are fed by previously learned POS tags and linguistic dependency trees (Lopez, 2008).

Table 3. Transformations learned by Brill's tagger from WSJ; adapted from (Brill, 1995).

Tag Transformations for Known Words				Tag Transformations for Unknown Words			
#	From	To	Condition	#	From	To	Condition
1	NN	VB	Previous tag is TO	1	NN	NNS	Has suffix -s
2	VBP	VB	One of the previous three tags is MD	2	NN	CD	Has character .
3	NN	VB	One of the previous two tags is MD	3	NN	JJ	Has character -
4	VB	NN	One of the previous two tags is DT	4	NN	VBN	Has suffix -ed
5	VBD	VBN	One of the previous three tags is VBZ	5	NN	VBG	Has suffix -ing

NLP learning begins with fundamental language notions. POS tags are frequently learned using ML algorithms. A widely used learning system in the public domain is the error-driven transformation-based Brill's tagger (Brill, 1995).¹⁸ The tagger uses the Penn Treebank POS tags. The learner reads an un-annotated text and automatically deduces POS labelling rules by comparison with a small manually annotated corpus. A labelling rule – a *transformation* – is judged by the number of errors produced on the annotated corpus. At each step, the algorithm chooses the transformation with the highest error reduction. Learning stops when no transformation can reduce errors against a pre-defined threshold. Table 3, adapted from (Brill, 1995), lists selected transformations learned on the Wall Street Journal corpus (Marcus et al, 1993); previously unseen words are tagged NN by default. For example, the NN → VB transformation replaces *to/TO reply/NN* with *to/TO reply/VB*.

The accuracy of POS taggers may increase sharply as the training data improve. Four HMM-based algorithms¹⁹ and Brill's tagger boosted their performance from 71.9%-77.2% on a lexicon with noisy, ambiguous tags to 93.9%-95.9% when no tag ambiguity was present (Banko and Moore, 2004). The authors clean out ambiguous tags by building *N-gram* models of the data. An *N-gram* is a sequence of items $w_1 \dots w_N$, accompanied by the probabilities of their occurrence in the corpus. In text processing, w_i is usually a word, whereas in speech processing it often denotes a letter or a sound. In practice, N seldom exceeds 3. *N-gram* models are computed as $P(w_k | w_1^{k-1}) \approx P(w_k | w_{k-N}^{k-1})$, where $P(w_k | w_{k-N}^{k-1})$ is the probability of the item w_k appearing after the sequence $w_{k-N} \dots w_{k-1}$. To keep unambiguous tags, Banko and Moore (2004) only selected trigrams in which all words had no more than one possible POS tag.

The results of *entity recognition*²⁰ are fed to learning problems in Text Summarization, Machine Translation, Question Answering and so on. Examples in Table 4, adapted from (Barzilay and Lapata, 2008), were used in an ML experiment with ranking document sentence permutations on their local coherence. In both excerpts, the underlined expressions refer to the same entity.

Table 4. Excerpts from (Barzilay and Lapata, 2008); one entity class is marked for each document.

Text Sample from Earthquakes	Text Sample from Accidents
The 7.3 quake hit <u>Menglian county</u> at 5:46 am. <u>The same area</u> was struck by a 6.2 tremor early Monday morning, the bureau said. The county is on the China-Burma border, and is a sparsely populated, mountainous region. The bureau's XuWei said some buildings sustained damage and there were some injuries, but he had no further details.	When <u>the pilot</u> failed to arrive for his brother's college graduation, concerned family members reported that he and his airplane were missing. A search was initiated, and the Civil Air Patrol located the airplane on top of Pine Mountain. According to <u>the pilot</u> 's flight log, the intended destination was Pensacola, FL, with intermediate stops for fuel at Thomson, GA, and Greenville, AL.

The entities were represented by grammatical features, e.g., their sentence roles (subject, object, other, absent). Further, these representations were used by learning algorithms: SVM, HMM and LSA. In this setting, SVM and LSA model only transitions between sentences. The algorithms do not have access to the document structure, so they focus on local coherence. In contrast, HMM has access to the document topics and their weighted order, so it is based on global coherence. Its states represent topics and state transitions – the probability of changing from one topic to another. The performance of the algorithms, measured by a fraction of correct pairwise document permutation ranking, varies across the collections. On Earthquakes and Accidents, respectively, the SVM-based algorithm achieved the accuracy of 87.2% and 90.4%, the HMM-based – 88.0% and 75.8%, the LSA-based – 81.0% and 87.3%. Both locally-focussed learners performed better on Accidents, where the vocabulary is rather restricted, and considerably outperformed the globally-oriented HMM-based learner. On Earthquakes, a collection with a richer vocabulary, the HMM-based learner outperformed the others, but not to the same extent.

Entity recognition is often applied to the analysis of information-rich texts such as newspaper and magazine articles. Its subfield, Named-Entity Recognition (NER), aims to find names which capture domain-specific information. Examples are protein, gene and drug names in biomedical texts (Kim et al, 2008), and person and geographic names in news reports (Steinberger and Pouliquen, 2007). For an example of multilingual NER, see (Ji and Grisham, 2008).

Sentence parsing establishes syntactic relations between words and groups of words. The results can assist sentence-level NLP tasks. Given a *grammar*, a parser outputs a tree-shaped sentence structure. Successful, correct and complete parse may be hard to achieve if the sentence has a complicated structure. That is partially because large-coverage, precise grammars of natural languages are highly recursive, and because proper connections between (more easily recognized) groups or phrases often require semantic information.

Rules-base parsers tend to be more linguistically accurate, but they are costly to build. An alternative parser-development technology is statistical parsing, which requires training data in the form of a grammatically annotated corpus. There are numerous parsers for English (in both styles), some of them in the public domain²¹, and they produce a rather baffling variety of structures. As an example, let us show a few parses of the sentence “*The boys played with the ball.*”

The Link Grammar parser (Sleator and Temperley, 1993) gives this tree²²:

```
( S
  ( NP The boys )
  ( VP played ( PP with ( NP the ball )))
  . )
```

Charniak’s statistical parser (Charniak, 2000) returns the following tree (the Stanford parser²³ produces an analogous structure):

```
[ S1 [ S
  [ NP [ DT [ The ] ] [ NNS [ boys ] ] ]
  [ VP [ VBD [ played ] ] ]
  [ PP [ IN [ with ] ] [ NP [ DT [ the ] ] [ NN [ ball ] ] ] ]
  [ . [ . ] ] ] ]
```

MiniPar²⁴ (a principle-based parser which ranks parse trees using Maximum Likelihood Estimation) produces this:

```
(U
(C (root fin)
(V played (root play) (relation i)
(N boys (root boy) (relation s)
(Det The (root the) (relation det)))
(N (root boy) (relation subj))
(Prep with (root with) (relation mod)
(N ball (root ball) (relation pcomp-n)
(Det the (root the) (relation det))))))
(U . (root .) (relation punc)))
```

In contrast with knowledge-rich parsing, the shallower process of *text chunking* divides a sentence into syntactically connected non-overlapping segments. For example, the same sentence “*The boys played with the ball.*” can be chunked as [NP *the boys*][VP *played*][PP *with*][NP *the ball*]. Text chunking often applies learning algorithms, for example, Winnow (Zhang et al, 2002) and SVM (Kudo and Matsumoto, 2003). Chunking may not be as beneficial for the quality of learning models as parsing based on a rigorously constructed grammar (Chan et al, 2007).

Deeper, and more ambitious, NLP tasks are concerned with the meaning of words, utterances, text segments. Such tasks often occupy an intermediate level of nested learning. The sense of a word can be learned from a combination of its morphological, syntactic and semantic characteristics, a local context of its use, and the topic of the text where it occurs (Lee and Ng, 2002).

Stevenson and Wilks (2001) present a system that disambiguates content words (nouns, verbs, adjectives and adverbs). The system first uses disambiguation modules built from information and knowledge sources: Brill’s tagger, a Named-Entity Recognizer, lexical lookup from Longman Dictionary of Contemporary English (LDOCE)²⁵. The modules’ results go into the memory-based learner TiMBL (Daelemans et al, 1999). Stevenson and Wilks (2001) evaluated the system on two resources of semantically tagged data, SEMCOR (Landes et al, 1998) and SENSUS (Knight and Luk, 1994). They showed that the interactive use of sources, including early removal of senses incorrect in the given context, gave significant disambiguation improvement over the baseline method which picked the most frequent sense in LDOCE. The improvement was 90.37% over 30.90% for the entire corpus. If calculated for word categories, the improvement was 91.24% over 34.56% for nouns, 88.38% over 18.46% for verbs, 91.09% over 25.76% for adjectives, and 70.61% over 36.73% for adverbs.

In some nested problems, ML applications first concentrate on particular parts of speech:

- adjectives and adverbs can be classified according to sentiments they bear (Benamara et al, 2007; Esuli and Sebastiani, 2006); the handbook chapter “Machine Learning in Mega-Text Processing” discusses the problem in more details;
- nouns can be identified as subjective or objective (Riloff et al, 2003), or as having positive or negative semantic orientation (Turney and Littman, 2003);

Table 5. Two meanings of the word *drive*; sentiment labels correspond to the meaning and to the Russian and French translation.

English	Sentiment	Meaning	Russian	French
<i>Drive into a village</i>	Neutral	Go by car, bus, truck	<i>Въехать в поселок</i>	<i>Entrer dans un village</i>
<i>Drive to despair</i>	Negative	Put someone in an emotional state	<i>Пригнать отчаяние</i>	<i>Mettre au désespoir</i>

- verbs are often classified based on Levin's classes of alteration (Levin, 1997); for more information on the use of semantic categories of verbs in general texts refer to (Lapata and Brew, 2004), in applications to biomedical domain – to (Korhonen et al, 2008).

The results of these three applications assist machine learning of utterance sentiment, opinions expressed in text segments and factual information extraction, respectively.

In the examples of NLP problems which we show in this section, getting the precise word meaning is essential. The next section discusses ML applications that undertake this ambitious task.

WORD SENSE DISAMBIGUATION

Word Sense Disambiguation is one of the core NLP tasks. Its success would be instrumental in Machine Translation, Multilingual Text Analysis, Sentiment Analysis, Question Answering, Information Extraction, and so on. In other words, most of the advanced problems in language processing require WSD in one capacity or another. For example, they rely on WSD in resolving *homonymy* (for example, whether *lie* means being prone or telling untruth). It is a challenging task to choose among different meanings of the same word. A popular example is the translation of *know* as French *savoir* or *connaître*. Table 5 shows a less known example: two meanings of the verb *drive*²⁶ define the sentiment labels of the text, and that affects its translations into Russian and French.

WSD can be considered a supervised learning task if a word is assigned a sense from a set of pre-defined labels. In WSD as an unsupervised task, word senses are grouped according to some criteria. For example, in a supervised setting, *drive* can have a set of labels travel, compel, push and so on. In *drive into a village*, an algorithm should label *drive* as travel, in *drive to despair* – as compel. In unsupervised leaning of sentiment, an algorithm can place *drive* in *drive to a village* and in *drive to despair* in two different clusters, one for sentiment-neutral expressions, and the other – for negative sentiment.

WSD as a supervised learning task requires training data, a repository of words annotated with senses. WordNet²⁷, a public-domain lexical knowledge base, is a very helpful resource, but its use requires much additional work. It represents – among many other lexical facts – word senses for English nouns, verbs, adjectives and adverbs. Highly polysemous words have literally dozens of fine-grained senses: for the verb *drive* WordNet lists twenty two!²⁸ The examples which we discussed earlier correspond to the following:

Sense 2: *drive, motor (travel or be transported in a vehicle)*

Sense 4: *force, drive, ram (force into or from an action or state, either physically or metaphorically)*

Sense 5: ***drive***(*to compel or force or urge relentlessly or exert coercive pressure on, or motivate strongly*).

The SemCor corpus²⁹ is a public-domain collection of 352 texts; 186 of the texts have all content words (nouns, verbs, adjectives, and adverbs) POS-, lemma- and sense-annotated; the remaining texts have verbs annotated with lemma and sense. All other words in all texts are POS-tagged. In addition, the named entities *persons*, *locations*, and *groups* are identified in SemCor. The characteristics make SemCor useful when the efficiency of a WSD system is important (Mihalcea and Csomai, 2005).

Roget's Thesaurus³⁰, which groups words together by implicit semantic relations, is another resource for WSD problems. It is worth noting that the WSD resources age well. For example, the 1911 and 1987 versions of Roget's do not differ significantly when used to determine semantic similarity or more generally semantic relatedness (Kennedy and Szpakowicz, 2008). The version with the 1911 data is publicly available.³¹

Triennial evaluations of WSD systems began in 1998.³² The evolution of SENSEVAL tasks over the years, the inclusion of languages other than English and a stiffer competition from the rapidly increasing number of participating systems amply demonstrate the many challenges that semantic analysis poses for ML algorithms. SENSEVAL-1 evaluated WSD in English, French and Italian. For English, 18 systems – 10 supervised and 8 unsupervised – were given a sample of words for which corresponding corpus instances were selected. The systems “competed” in 41 disambiguation tasks, for 15 nouns, 13 verbs, 8 adjectives and 5 words not POS-tagged (Kilgarriff and Rosenzweig, 2000).

SENSEVAL-2 added Chinese, Czech, Danish, Dutch, Spanish, Swedish, Estonian, Japanese, Korean and Basque (Edmonds, 2002). In English SENSEVAL-2, 26 systems participated in the lexical sample task, where 73 words had their POS predetermined. In a new all-word disambiguation task, 21 systems labelled all content words in a text. One, perhaps unintended, result of SENSEVAL-2 was a comprehensive discussion of supervised learners: three Bayesian algorithms, decision lists and a transformation-based learner. Yarowsky and Florian (2002) compared the learners' disambiguation performance with respect to 13 parameters. Let us list the parameters to show the multi-dimensional nature of search for word's meaning: 1) target language; 2) part of speech; 3) sense granularity; 4) inclusion and exclusion of major feature classes; 5) variable context width; 6) number of training examples; 7) baseline probability of the most likely sense; 8) sense distributional entropy; 9) number of senses per keyword; 10) divergence between training and test data; 11) degree of (artificially introduced) noise in the training data; 12) the effectiveness of an algorithm's confidence rankings; and 13) a full keyword breakdown of the performance of each algorithm (Yarowsky and Florian, 2002).

English SENSEVAL-3 compared the systems' performance on lexical sample semantic labelling (20 nouns, 32 verbs, 5 adjectives) and identification of multi-word expression (Mihalcea et al, 2004), all-word disambiguation task and multi-lingual lexical sample task (Chinese-English). 47 systems participated in the lexical sample task. Supervised systems significantly improved over the most-frequent-sense baseline. SEMEVAL-1/SENSEVAL-4, held in 2007, had 18 tasks, some of them general (e.g., cross-language information retrieval, lexical substitution or metonymy resolution), some – restricted (e.g., sense tagging of affective text, disambiguation of prepositions, classification of relations between nominals). English was present in 16 tasks. The number of participating systems exceeded 90.

Choices made among many WSD options affect learning results. We illustrate this on three specific WSD challenges: disambiguation of infrequent words, word representation from a given context and coping with domain-specific information.

Table 6. The best micro-averaged Recall (%) among SVM, NB, DT, and AdaBoost on word-sense disambiguation of Senseval-2 and Senseval-1 data; no statistical feature selection performed.

SENSEVAL-1										
i		ii		iii		iv		v		
NB	71.6	SVM	70.3	SVM	74.0 <th>NB</th> <td>70.4</td> <th>SVM</th> <td>79.2</td> <th></th>	NB	70.4	SVM	79.2	
SENSEVAL-2										
i		ii		iii		iv		v		
NB	58.0	SVM	57.7	SVM	60.5 <th>SVM</th> <td>54.5</td> <th>SVM</th> <td>65.4</td> <th></th>	SVM	54.5	SVM	65.4	

Disambiguation of low-frequency words can be a bottleneck for WSD systems because infrequent words are hard to find in annotated data. Shirai and Yagi (2004) suggested combining two learners. One learner disambiguates high-frequency words in a supervised setting, from annotated data, another – low frequency words by extraction of pre-defined information from word definition sentences. The authors used SVM as a supervised learner and NB to model the word's sense based on the extracted information. The system was tested on a corpus of approximately 200000 Japanese sentences extracted from newspaper and magazine articles. NB derived the information – in this case hypernyms – from an electronic concept dictionary of Japanese. Using the ensemble of two classifiers allowed an F-score increase compared with separate disambiguation: 70.13% (the ensemble) versus 67.04% (only SVM) and 64.06% (only NB).

The following example shows how the performance of an algorithm depends on word representation. Let us consider supervised learning of senses in Senseval-1 and Senseval-2 English data sets (Lee and Ng, 2002). The target words were represented in five ways: (i) part-of-speech of seven words (the word w in question, three neighbouring words to the left and right); (ii) the presence of words in the context surrounding w (all words in the context were replaced by their morphological root, stop words³³ were removed, all remaining words contributed one feature, a word could be in a sentence different than w); (iii) local collocations (the words be in the same sentence as w , all stop words retained); (iv) syntactic relations resulted from sentence parsing; (v) a combination of all the features (i) to (iv). For representations (ii)-(v), there were two options: only words which appear three or more times were selected (FS), and no feature selection was performed (No FS). Four learning algorithms were applied: SVM, NB, DT, and AdaBoost (it trains an ensemble of weak learners by providing more weights to misclassified examples). The micro-average Recall shows that every algorithm except AdaBoost performed better

Table 7. The best micro-averaged Recall (%) among SVM, NB, DT, and AdaBoost on word-sense disambiguation of Senseval-2 and Senseval-1 data; features selected according to word frequency.

SENSEVAL-1							
ii		iii		iv		v	
NB	67.3	NB	70.3	NB	69.8	DT	77.7
SENSEVAL-2							
ii		iii		iv		v	
NB	55.8	DT	57.2	DT	54.2	NB	62.7

than others on some feature sets, but AdaBoost was the second best on most of the tasks. Tables 6 and 7, adapted from (Lee and Ng, 2002), list the results.

ML methods are often applied to text data mining of knowledge- and terminology-rich texts – biomedical, geographical, political and so on. To conduct proper learning, algorithms need access to domain-specific WSD, but the general-purpose resources such as WordNet may not include domain-specific meanings of a word. For example, the word *dialog* is marked with three senses which exclude its use in man-machine interaction³⁴:

Sense 1: *a conversation between two persons*;

Sense 2: *the lines spoken by characters in drama or fiction*;

Sense 3: *a literary composition in the form of a conversation between two people*.

Similar challenges can arise in the study of biomedical texts. For the word *acne*, WordNet lists one sense: *an inflammatory disease involving the sebaceous glands of the skin; characterized by papules or pustules or comedones*³⁵. This misses a possible response to a contact with irritating substances (Hecht and Shiel, 2003). To fill the gap, Lin and Pantel have designed the algorithm of Clustering By Committee (CBC). For each word, it learns a set of concepts represented in texts. The algorithm was applied to a news corpus to find domain-specific noun senses (Lin and Pantel, 2002). On two test sets it outperformed seven clustering algorithms in terms of cluster consistency with an assumed answer key. On a set of 13403 words, CBC achieved 60.95% consistency, whereas the second result by K-means was 56.70%. On another set of 3566 words, CBC achieved 65.82% compared with the next-best result 63.15% by Buckshot, a hybrid clustering algorithm which combines partitioning and hierarchy (Cutting et al, 1992). This work also illustrates well the position of WSD in nested NLP machine learning applications. The authors designed an unsupervised learning algorithm of Discovering Inference Rules from Texts (DIRT). Their goal was to enforce Question Answering with an algorithm capable of text-based inference, e.g. automatically finding such rules as a rather trivial “*X wrote Y* ≈ *X is the author of Y*” and more elaborate “*X manufactures Y* ≈ *X’s Y factory*” (Lin and Pantel, 2001). DIRT outputs a set of associations between paths in dependency trees. To obtain these linguistic trees, texts are parsed with MiniPar (Lin, 1998). Its lexicon was derived from parts of speech and subcategorization frames used by WordNet.

DISCUSSION

This chapter has shown how Natural Language Processing applications use Machine Learning algorithms to solve typical language tasks. Our choice of ML applications has been necessarily limited by the chapter size. We have left out many applications popular among NLP researchers and practitioners. Let us consider an impressive example of deep learning architecture (Hinton et al., 2006). Deep Neural Networks – with several hidden layers – were designed to tackle complex language problems by combining semantic and probabilistic models (Bengio et al., 2001). Later, the algorithm has been used in feature learning for a joint solution of several NLP tasks (Collobert and Weston, 2008). Collobert and Weston divided the problem of finding proper text features into six sub-problems: part-of-speech tagging, chunking, Named-Entity Recognition, semantic role labeling, real language models (labelling real text as positive examples and generating artificial text as negative examples), and finding synonyms. This multi-task learning works well for frequent words and for rare words. It identifies *Spain*, *Italy*, *Russia* as

semantically related to *France* (a frequent word), and *smashed, ripped, brushed* as semantically related to *scratched* (a rare word). In future, deep architecture algorithms, not necessarily confined to Neural Nets, can be expected to learn complex semantic tasks when only limited prior knowledge is available and successfully operate in the absence of handcrafted features. The algorithms are expected to learn complex tasks for which training data is scarce; they will build abstractions with little human input and bring a considerable progress into unsupervised learning, among other advantages for the future of Machine Learning (Bengio, 2009).

Let us close by offering a few simple suggestions regarding the choice of ML algorithms for NLP problems. Before one invests considerable time and effort in ML applications, a few questions on the task and on data characteristics are worth asking. Is it corpus analysis or text analysis? In corpus analysis, a data set is considered as one collection of text (Kilgarriff, 2001). The analysis, usually statistical and automated, applies to the whole collection, without dividing it into separate texts. ML applications very commonly employ N -grams as a means of corpus analysis. Is it a supervised or unsupervised problem? If it is a supervised learning problem, is there enough annotated data for training and testing? Will it be a form of topic classification? (The answer may affect the choice of feature selection methods.) Next, one ought to settle on the unit of classification: will words, text segments or complete texts be classified? Finally, the quality of data has to be considered; for example, a large number of spelling and grammar errors may require more robust learners.

ABBREVIATIONS

- DT: Decision Trees
- HMM: Hidden Markov Model
- IE: Information Extraction
- kNN: k-Nearest Neighbor
- LSA: Latent Semantic Analysis
- ML: Machine Learning
- NB: Naïve Bayes
- NER: Named-Entity Recognition
- NLP: Natural Language Processing
- POS: Part-of-Speech
- SNLP: Statistical Natural Language Processing
- SVM: Support Vector Machine
- WSD: Word Sense Disambiguation

ACKNOWLEDGEMENT

This work has been partially supported by the Ontario Centres of Excellence and by the Natural Sciences and Engineering Research Council of Canada.

REFERENCES

- Agirre, E., & Martinez, D. (2001). *Knowledge sources for word sense disambiguation*. Paper presented at the 4th International Conference on Text, Speech and Dialogue
- Banko, M., & Moore, R. (2004). *Part of speech tagging in context*. Paper presented at the 20th International Conference on Computational Linguistics (Coling 2004).
- Barzilay, R., & Lapata, M. (2008). Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1), 1–34. doi:10.1162/coli.2008.34.1.1
- Benamara, F., Cesarano, C., Picariello, A., Reforgiato, D., & Subrahmanian, V. (2007). *Sentiment analysis: Adjectives and adverbs are better than the adjectives alone*. Paper presented at the International Conference on Weblogs and Social Media (ICWSM’2007).
- Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 1(1).
- Bengio, Y., Ducharme, R., & Vincent, P. (2001). *A neural probabilistic language model*. Paper presented at the 13th conference on Neural Information Processing Systems (NIPS 2000).
- Biber, D. (1988). Variation across speech and writing. In D. Biber (Ed.), *Variation across speech and writing* (pp. 3-27). Cambridge, UK: Cambridge University Press.
- Blatak, J., Mrakova, E., & Popelinsky, L. (2004). *Fragments and text classification*. Paper presented at the Association for Computational Linguistics (ACL-2004).
- Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging . *Computational Linguistics*, 21(4), 543–565.
- Chan, Y., Ng, H., & Chiang, D. (2007). *Word sense disambiguation improves statistical machine translation*. Paper presented at the Association for Computational Linguistics (ACL-2007).
- Charniak, E. (2000). *A maximum entropy inspired parser*. Paper presented at the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2000).
- Choi, Y., Cardie, C., Riloff, E., & Patwardhan, S. (2005). *Identifying sources of opinions with conditional random fields and extraction patterns*. Paper presented at the Empirical Methods for Natural Language Processing (EMNLP 2005).
- Colas, F., & Brazdil, P. (2006). Comparison of SVM and some older classification algorithms in text classification tasks. In M. Bramer (Ed.), *Artificial intelligence in theory and practice* (pp. 169-178). Berlin, Germnay: Springer.
- Collins, M., Hajic, J., Ramshaw, L., & Tillmann, C. (1999). *A statistical parser for Czech*. Paper presented at the Association for Computational Linguistics (ACL-99).
- Collobert, R., & Weston, J. (2008). *A unified architecture for natural language processing: Deep neural networks with multitask learning*. Paper presented at the 25th International Conference on Machine Learning (ICML-2008).

- Crystal, D. (2006). *Language and the Internet*. Cambridge, UK: Cambridge University Press.
- Cutting, D., Karger, D., Pedersen, J., & Turkey, J. (1992). *A cluster-based approach to browsing large document collections*. Paper presented at the Annual International Conference of Special Interest Group on Information Retrieval (SIGIR-92).
- Daelemans, W. (2006). *A mission for computational natural language learning*. Paper presented at the 10th Conference on Computational Natural Language Learning (CoNLL-X).
- Daelemans, W., Zavrel, J., van der Sloot, K., & van der Bosch, A. (1999). *TiMBL: Tilburg memory base learner, version 2.0, reference guide* (Tech. Rep.). The Netherlands: University of Tilburg.
- Dredze, M., Blitzer, J., & Pereira, F. (2007). *Biographies, Bollywood, boom-boxes, and blenders: Domain adaptation for sentiment classification*. Paper presented at the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007).
- Edmonds, P. (2002). Introduction to Senseval. *ELRA Newsletter*, 7(3).
- Esuli, A., & Sebastiani, F. (2006). *Determining term subjectivity and term orientation for opinion mining*. Paper presented at the European Chapter of the Association for Computational Linguistics (EACL'07).
- Feldman, R., & Sanger, J. (2007). *The text mining handbook: Advanced approaches in analyzing unstructured data*. Cambridge, UK: Cambridge University Press.
- Firth, J. (1936). Alphabets and phonology in India and Burma. *Bulletin of the School of Oriental Studies, University of London*, 8, 517–546.
- Firth, J. (1957). A synopsis of linguistic theory 1930-1955. In *Studies in Linguistic Analysis* (pp. 1 -32). Oxford: Basil Blackwell.
- Gabrilovich, E., & Markovitch, S. (2004). Text categorization with many redundant features: Using aggressive feature selection to make SVMs competitive with C4.5. In *Proceedings of the International Conference on Machine Learning ICML 2004*.
- Grenader, T., Klein, D., & Manning, C. (2005). *Unsupervised learning of field segmentation models for information extraction*. Paper presented at the 43rd Annual Meeting on Association for Computational Linguistics.
- Harris, Z. (1968). *Mathematical structures of language*. New York: Interscience Publishers.
- Hecht, F., & Shiel, W. (2003). *Webster's new world medical dictionary* (2nd ed.). New York: John Wiley & Sons Publishing.
- Hinton, G., Osindero, S., & Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18, 1527–1554. doi:10.1162/neco.2006.18.7.1527
- Hu, M., & Liu, B. (2004). *Mining and summarizing customer reviews*. Paper presented at the International Conference on Knowledge Discovery and Data Mining (KDD'04).

- Ji, H., & Grisham, R. (2008). *Refining event extraction through unsupervised cross-document inference*. Paper presented at the Annual Meeting of the Association of Computational Linguistics.
- Joachims, T. (2002). *Learning to classify text using support vector machines - methods, theory, and algorithms*. Amsterdam: Kluwer Academic Publishers.
- Jurafsky, D., & Martin, J. (2008). *Speech and language processing: An introduction to natural language processing, computational linguistics and speech recognition* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.
- Kennedy, A., & Szpakowicz, S. (2008). *Evaluating Roget's thesauri*. Paper presented at the Association for Computational Linguistics (ACL 2008).
- Kilgarriff, A. (2001). Comparing corpora. *International Journal of Corpus Linguistics*, 6(1), 97–133. doi:10.1075/ijcl.6.1.05kil
- Kilgarriff, A., & Rosenzweig, J. (2000). Framework and results for English SENSEVAL. *Computers and the Humanities*, 34(1/2), 15–48. doi:10.1023/A:1002693207386
- Kim, J.-D., Ohta, T., & Tsujii, J. (2008). Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*, 9(10), e25.
- Knight, K., & Luk, S. (1994). *Building a large knowledge base for machine translation*. Paper presented at the American Association for Artificial Intelligence Conference (AAAI-94)
- Korhonen, A., Krymolowski, Y., & Collier, N. (2008). *The Choice of Features for Classification of Verbs in Biomedical Texts*. Paper presented at the International Conference on Computational Linguistics (COLING-2008).
- Kucera, H., Francis, W., & Carroll, J. (1967). *Computational analysis of present-day American English*. Providence, RI: Brown University Press.
- Kudo, T., & Matsumoto, Y. (2003). *Fast methods for kernel-based text analysis*. Paper presented at the 41st Annual Meeting on Association for Computational Linguistics.
- Lai, C.-C., & Tsai, M.-C. (2004). *An empirical performance comparison of machine learning methods for spam e-mail categorization*. Paper presented at the Fourth International Conference on Hybrid Intelligent Systems (HIS'04).
- Landauer, T., & Dumais, S. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104(2), 211–240. doi:10.1037/0033-295X.104.2.211
- Landes, S., Leacock, C., & Tengi, R. (1998). Building a semantic concordance of English. In C. Fellbaum (Ed.), *WordNet: An electronic lexical database and some applications*. Cambridge, MA: MIT Press.
- Lapata, M., & Brew, C. (2004). Verb class disambiguation using informative priors. *Computational Linguistics*, 30(1), 45–73. doi:10.1162/089120104773633385
- Lee, M., Wang, W., & Yu, H. (2006). Exploring supervised and unsupervised methods to detect topics in biomedical text. *BMC Bioinformatics*, 7(140).

- Lee, Y., & Ng, H. (2002). *An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation*. Paper presented at the Conference on Empirical Methods in Natural Language Processing (EMNLP-2002).
- Leech, G., Garside, R., & Bryant, M. (1994). *CLAWS4: The tagging of the British National Corpus*. Paper presented at the 15th International Conference on Computational Linguistics (COLING 94).
- Levin, B. (1993). *English verb classes and alterations: A preliminary investigation*. Chicago, IL: The Chicago University Press.
- Lin, D. (1998). *Dependency-based evaluation of MINIPAR*. Paper presented at the Workshop on the Evaluation of Parsing Systems.
- Lin, D., & Pantel, P. (2001). Discovery of inference rules for question answering. *Natural Language Engineering*, 7(4), 343–360. doi:10.1017/S1351324901002765
- Lin, D., & Pantel, P. (2002). *Concept discovery from text*. Paper presented at the Conference on Computational Linguistics (COLING-02).
- Lopez, A. (2008). Statistical machine translation. *ACM Computing Surveys*, 40(3). doi:10.1145/1380584.1380586
- Manning, C., & Schütze, H. (2003). *Foundations of statistical natural language processing* (6th ed.). Cambridge, MA: The MIT Press.
- Marcus, M., Santorini, B., & Marcinkiewicz, M. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2), 313–330.
- Mihalcea, R., & Csomai, A. (2005). *SenseLearner: Word sense disambiguation for all words in unrestricted text*. Paper presented at the Annual Meeting of the Association for Computational Linguistics (ACL-2005).
- Mitchell, T. (1997). *Machine learning*. New York: McGraw-Hill.
- Mitkov, R. (Ed.). (2003). *The Oxford handbook of computational linguistics*. Oxford, UK: Oxford University Press.
- Miyao, Y., Sætre, R., Sagae, K., Matsuzaki, T., & Tsujii, J. (2008). *Task-oriented evaluation of syntactic parsers and their representations*. Paper presented at the Association for Computational Linguistics.
- Och, F. (2005). *Statistical machine translation: Foundations and recent advances*. Paper presented at the Machine Translation Summit.
- Popescu, O., & Magnini, B. (2007). *Sense discriminative patterns for word sense disambiguation*. Paper presented at the Workshop on Semantic Content Acquisition and Representation (SCAR).
- Porter, M. (1980). An algorithm for suffix stripping. *Program*, 14(3), 130–137.
- Rijsbergen, K. van. (1979). *Information retrieval*. London: Butterworths.
- Riloff, E., Wiebe, J., & Wilson, T. (2003). *Learning subjective nouns using extraction pattern bootstrapping*. Paper presented at the 7th Conference on Natural Language Learning (CONLL-03).

- Samuelsson, C. (2004). Statistical methods. In R. Mitkov (Ed.), *The Oxford handbook of computational linguistics* (pp. 358–375). Oxford, UK: Oxford University Press.
- Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1), 1–47. doi:10.1145/505282.505283
- Shawe-Taylor, J., & Christianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge, UK: Cambridge University Press.
- Sleator, D., & Temperley, D. (1993). *Parsing English with a link grammar*. Paper presented at the Third International Workshop on Parsing Technologies.
- Sokolova, M., & Lapalme, G. (2007). *Performance measures in classification of human communication*. Paper presented at the 20th Canadian Conference on Artificial Intelligence (AI'2007).
- Sokolova, M., Nastase, V., & Szpakowicz, S. (2008). *The telling tail: Signals of success in electronic negotiation texts*. Paper presented at the Third International Joint Conference on Natural Language Processing (IJCNLP 2008).
- Steinberger, R., & Pouliquen, B. (2007). Cross-lingual named entity recognition. *Linguisticae Investigationes*, 30(1), 135–162.
- Stenning, K., Lascarides, A., & Calder, J. (2006). *Introduction to cognition and communication*. Cambridge, MA: The MIT Press.
- Stevenson, M., & Wilks, Y. (2001). The interaction of knowledge sources in word sense disambiguation. *Computational Linguistics*, 27(3), 321–349. doi:10.1162/089120101317066104
- Thomas, M., Pang, B., & Lee, L. (2006). *Get out the vote: Determining support or opposition from congressional floor-debate transcripts*. Paper presented at the Empirical Methods in Natural Language Processing (EMNLP 2006).
- Turney, P., & Littman, M. (2003). Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4), 315–346. doi:10.1145/944012.944013
- Varasai, P., Pechsiri, C., Sukvari, T., Satayamas, V., & Kawtrakul, A. (2008). *Building an annotated corpus for text summarization and question answering*. Paper presented at the Sixth International Language Resources and Evaluation (LREC'08).
- Wiebe, J., & Riloff, E. (2005). *Creating subjective and objective sentence classifiers from unannotated texts*. Paper presented at the Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2005).
- Witte, R., & Bergler, S. (2007). *Fuzzy clustering for topic analysis and summarization of document collections*. Paper presented at the 20th Canadian Conference on Artificial Intelligence.
- Wolpert, D., & Macready, W. (2005). Coevolutionary free lunches. *IEEE Transactions on Evolutionary Computation*, 9(6), 721–735. doi:10.1109/TEVC.2005.856205
- Yarowsky, D., & Florian, R. (2002). Evaluating sense disambiguation across diverse parameter spaces. *Natural Language Engineering*, 8(4), 293–310. doi:10.1017/S135132490200298X

Zhang, T., Damerau, F., & Johnson, D. (2002). Text chunking based on a generalization of Winnow. *Journal of Machine Learning Research*, 2, 615–637. doi:10.1162/153244302320884560

Zipf, G. (1935). *The psycho-biology of language*. Boston: Houghton Mifflin.

KEY TERMS AND DEFINITIONS

Corpus: an organized collection of texts.

Entity Recognition: the task/process of finding in text expressions (usually nouns and noun phrases) that serve as names of entities such as people, organizations, places, dates and so on

Natural Language Processing: theory, design and implementation of systems for the analysis, understanding and generation of written or spoken language

Nested Learning: learning based on the results of other learning

Parsing: the process of producing a linguistic structure, usually syntactic (for clauses or sentences), sometimes morphological (for words, especially in inflected languages)

Part-of-Speech Tagging: the process of labelling each word in a corpus with its part of speech (much more detailed than the traditional *noun*, *verb*, *adjective* and so on)

Word-Sense Disambiguation: the task/process of selecting the correct sense of a word (usually in context).

ENDNOTES

¹ <http://www.aclweb.org/>

² Not all NLP algorithms that use annotated data are presented as supervised. A system which we would consider supervised may be described otherwise.

³ <http://tartarus.org/~martin/PorterStemmer/>

⁴ The stemmer was later applied to Indo-European languages – a few Romance, a few Germanic, plus Russian – and to Finnish of the Finno-Ugric group (snowball.tartarus.org/texts/stemmersoverview.html).

⁵ $tf_{i,j}$ is the (normalised) frequency of term i in document j. idf_i is the (natural logarithm of the) inverse document frequency – the total number of documents divided by the number of documents that contain term i. The score is high if a term is frequent in few documents but rare overall.

⁶ <http://people.csail.mit.edu/jrennie/20Newsgroups/>

⁷ <http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html>

⁸ <http://www.nlm.nih.gov/mesh/>

⁹ <http://world-gazetteer.com/>

¹⁰ <http://biocaster.nii.ac.jp/index.php?page=ontology&type=quick>

¹¹ <http://www.cis.upenn.edu/~treebank/> and <http://www.ldc.upenn.edu/LDC/online/treebank/>

¹² <http://ufal.mff.cuni.cz/pdt/>

¹³ <http://www-tsujii.is.s.u-tokyo.ac.jp/GENIA/home/wiki.cgi?page=GENIA+Treebank>

¹⁴ <http://www.natcorp.ox.ac.uk/>

¹⁵ <http://icame.uib.no/brown/bcm.html>

- 16 <http://crl.ucsd.edu/corpora/>
- 17 The term has several uses in pedagogy, but has not been applied to ML.
- 18 <http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/nlp/parsing/taggers;brill/>
- 19 HMM can also be used in an unsupervised setting (Grenader, 2005).
- 20 Other terms are *entity identification* and *entity extraction*.
- 21 <http://www.informatics.susx.ac.uk/research/groups/nlp/rasp/> is one of the newer additions.
- 22 <http://www.link.cs.cmu.edu/link/>
- 23 <http://nlp.stanford.edu:8080/parser/>
- 24 <http://www.cs.ualberta.ca/~lindek/minipar.htm>
- 25 <http://www.ldoceonline.com/>
- 26 Longman Dictionary of Contemporary English (<http://www.ldoceonline.com/>)
- 27 <http://wordnet.princeton.edu/>
- 28 Recorded Nov. 18, 2008, at 3:00 pm EST. Such granularity “beats” many dictionaries: Longman Dictionary of Contemporary English assigns thirteen verb senses to *drive*.
- 29 <http://multisemcor.itc.it/semcor.php>
- 30 <http://thesaurus.reference.com/>
- 31 <http://rogets.site.uottawa.ca/>
- 32 <http://www.senseval.org/>
- 33 Those are usually articles, conjunctions, prepositions and other function words, which appear very frequently in English texts of all lengths and genres.
- 34 Recorded Dec 11th, 2008, at 1:45 pm EST.
- 35 Recorded Dec 11th, 2008, at 2:10 pm EST.

Chapter 15

Machine Learning Applications in Mega-Text Processing

Marina Sokolova
CHEO Research Institute, Canada

Stan Szpakowicz
University of Ottawa, Canada and Polish Academy of Sciences, Poland

ABSTRACT

This chapter presents applications of machine learning techniques to problems in natural language processing that require work with very large amounts of text. Such problems came into focus after the Internet and other computer-based environments acquired the status of the prime medium for text delivery and exchange. In all cases which the authors discuss, an algorithm has ensured a meaningful result, be it the knowledge of consumer opinions, the protection of personal information or the selection of news reports. The chapter covers elements of opinion mining, news monitoring and privacy protection, and, in parallel, discusses text representation, feature selection, and word category and text classification problems. The applications presented here combine scientific interest and significant economic potential.

INTRODUCTION

The chapter presents applications of Machine Learning (ML) to problems which involve processing of large amounts of texts. Problems best served by ML came into focus after the Internet and other computer-based environments acquired the status of the prime medium for text delivery and exchange. That is when the ability to work extremely large amounts of texts, which ML applications had not previously faced, became a major issue. The resulting set of techniques and practices, which we name *mega-text language processing*, are meant to deal with a mass of informally written, loosely edited text. A case in point is the analysis of opinions expressed in short informal texts written and put on the Web by the general public (Liu 2006). The sheer volume and variety of suddenly available language data has neces-

sarily invited the use of computing software capable of handling such a mass of data, learning from it and acquiring new information.

Until now, no clearly delineated subfield of Natural Language Processing (NLP) dealt with mega-texts – textual data on the Web, computer-mediated text repositories and in general texts in electronic format. Text Data Mining – a form of Data Mining – concerns itself with deriving new information from texts, but most often restrains from the study of language. Still, many researchers focus on the study of language, for example lexical, grammar and style issues, in such texts (Crystal 2006; Liu 2006). That no overarching NLP discipline has emerged can be explained by the fact that electronic texts and old-fashioned texts in books or newspapers share major characteristics. We discuss these characteristics in the handbook chapter “Machine Learning in Natural Language Processing”.

This chapter will show that ML techniques measure up well to the challenges that mega-texts pose. We focus on applications in aid of the study of language. In all cases which we discuss, an algorithm has ensured a meaningful result, be it the knowledge of consumer opinions, the protection of personal information or the selection of news reports. Although we mostly focus in this chapter on text classification problems, we go beyond document topic classification. English, the most popular language of the Web, is the default language of much of the scientific discourse. We state when problems deal with languages other than English.

In the chapter we cite standard measures used in NLP (Precision, Recall, F-score). Calculated for classifiers produced by an algorithm, they build on the numbers of correctly classified positive examples TP , incorrectly classified positive examples FP , and incorrectly classified negative examples FN .

$$\text{Precision: } P = \frac{TP}{TP + FP} \quad (1)$$

$$\text{Recall: } R = \frac{TP}{TP + FN} \quad (2)$$

F-score is a weighted sum of Precision and Recall:

$$F = \frac{(\beta^2 + 1)P}{(\beta^2 + 1)P + R} \quad (3)$$

In some cases authors use the traditional Accuracy, which we cite:

$$A = \frac{TP + TN}{TP + TN + FP + FN} \quad (4)$$

The remainder of the chapter is organized in four sections and some supplementary material. First, in three consecutive sections, we discuss ML applications in specific NLP mega-text problems. We concentrate on the correspondence between NLP problems and ML algorithms. The NLP problems arise from the need to analyze a high volume of electronic texts in marketing, mass media, and health care. Each problem is well served by different algorithms. Opinion analysis, originally a marketing problem, is mature enough to have become almost a traditional task in text analysis. On the other hand, privacy protection, especially urgent in health care, is the subject of cutting-edge research at the boundary of

NLP and ML. For each problem, we discuss the learning task, challenges posed by data, language characteristics, and results of experiments. We pay attention to different data representations and to the contribution of the differences to algorithm performance. The last section contains several concluding remarks and a few practical recommendations. A list of abbreviations used, a large bibliography, and key terms close the chapter. Note that wherever the discussion requires it, we refer the reader to the material in the handbook chapter “Machine Learning in Natural Language Processing”.

This chapter makes a detailed and thorough comparison of the performance of ML algorithms that help solve knowledge-intensive NLP problems where mega-texts are a major factor. We present applications which combine scientific interest and high economic potential.

OPINION MINING

Introduction

There is a recent trend for Web sites (such as Epinions.com or Amazon.com) to invite the general public to post their opinions on different types of products, services, social events and political events. Data gathered from such posts have become an important source of information about personal preferences, product perception and emerging public trends, among others. A typical data set of customer-written product reviews is a large number of loosely structured, mostly unedited short texts. The textual characteristics of the data and their volume strongly suggest the application of ML methods.

The common research initiatives in NLP and ML geared toward studying opinions and other subjective traits are usually referred to as *sentiment analysis*. It is a broad term, allowing for simple polarity, graded preferences and phenomena as complex as emotions. Numerous studies have been devoted to the assessment of subjectivity in texts and sentiment analysis (Liu 2006). In this chapter, we only have room for several references relevant to our themes; the current literature merits a separate handbook.

We focus on *opinion mining*, a subfield of sentiment analysis, concerned with finding opinions in texts. An opinion is an idea or belief about a particular subject. It is more focused than sentiment, which also includes one’s feelings and emotions. In its current form, opinion mining asks whether a term, a text segment or a complete text bears a positive or negative opinion (Kim and Hovy, 2004). NLP and ML researchers pay this area a lot of attention, in part because of a possible funding appeal (companies, government agencies and non-profit organizations may be willing to sponsor research), in part because so much data is readily available. The research has found application far beyond marketing. Opinion mining has been applied to political studies, for example prediction of election results (Kim and Hovy, 2007), and to spam filtering (Jindal and Liu, 2008). The granularity varies from a pair of words to paragraphs covering certain topic to complete texts (Turney and Littman, 2003; Breck et al, 2007; Jindal and Liu, 2008).

An extensive overreach of opinion mining problems has resulted in immensely diverse data quality. Opinion can be voiced by a professional critic or journalist (Wilson et al, 2006) or a politician (Thomas et al, 2006). Such texts usually come with good writing style, grammar and spelling. In most of the cases, however, data contributors are not professionals, so they do not necessarily follow rigid editorial rules and regulations. As a result, texts have grammatical and spelling errors and style deviations (Hu and Liu, 2004; Kim and Hovy, 2007; Jindal and Liu, 2008). Figure 1 presents excerpts from texts with various degrees of editing.

Figure 1. Excerpts from opinion mining data sets. The first sample comes from the Foreign Broadcast Information Service (FBIS), a US government agency (Wiebe and Riloff, 2005), the middle sample – from an election prediction message board (Kim and Hovy, 2007), the last sample – from custom-written product reviews (Hu and Liu, 2004).

<i>They are no more than frail excuses and pretexts to evade the peace process since this process does not agree with the ideology of expansion and the building of settlement.</i>
<i>The Liberal will win a close one. The rural polls in Wascana will vote overwhelmingly for the Alliance Party but the NDP will not siphon away enough votes from Liberals in the urban polls to help the Alliance win.</i>
<i>The strap is horrible and gets in the way of parts of the camera you need access to.</i>

Problem Description

The core of opinion mining is based on the assessment of the objective/subjective orientation of a word and its use in the context of the overall expression of an opinion (Esuli and Sebastiani, 2007). To borrow examples from (Esuli and Sebastiani, 2006a): *honest* and *disturbing* are subjective, whereas *triangular* is not. Another subfield of opinion mining is a study of positive and negative opinions based on the polarity of subjective opinion-bearing words (e.g., positive *nice* and negative *nasty* (Turney and Littman, 2003)) and their combinations (e.g., *very very good* conveys a strong opinion and *possibly less expensive* conveys a weak opinion (Benamara et al, 2007)). Word opinion tags can be extracted from electronic general lexical resources (in (Andreevskaia and Bergler, 2006) the authors used WordNet (Fellbaum, 1998) to assign sentiment categories to adjectives), and electronic sources of sentiment and emotion tags (such as WordNet Affect (Strapparava et al, 2006) or Linguistic Inquiry and Word Count (Pennebaker et al, 2001)).

In language, the use of adjectives, adverbs, negations and modal verbs often signals opinions. The handbook chapter “Machine Learning in Natural Language Processing” covers in more detail *part-of-speech* (POS) tagging. It is natural to see that many studies apply the POS tagging as the first step of

Table 1. Examples of objective and subjective sentences and their POS tagging, adopted from (Huang 2006). The used POS tags com from the Penn Treebank (Marcus et al, 1993). CD means cardinal number, DT – determiner, IN – preposition or subordinate conjunction, JJ – adjective, NN – noun, singular or mass, NNS plural noun, TO – to, VB – verb, base form, VBZ – verb, third-person-singular, present, WP – wh-pronoun.

Subjectivity	Sentence	POS tagging
Subjective	<i>The script is a tired one, with few moments of joy rising above the stale material</i>	<i>The/DT script/NN is/VBZ a/DT tired/JJ one/CD, /, with/IN few/JJ moments/NNS of/IN joy/NN rising/VBG above/IN the/DT stale/JJ material/NN</i>
Objective	<i>David is a painter with painter's block who takes a job as a waiter to get some inspiration</i>	<i>David/NN is/VBZ a/DT painter/NN with/IN painter/NN 's/POS block/NN who/WP takes/VBZ a/DT job/NN as/IN a/DT waiter/NN to/TO get/VB some/DT inspiration/NN</i>

Table 2. Binary positive/negative classification accuracy (%) of movie review data, adapted from (Boiy et al, 2007).

Algorithms	Movie Reviews		
	Unigrams	Bigrams	Adjectives
NBM	81.45%	83.15%	82.00%
SVM	85.45%	85.35%	75.85%
MaxEntropy	84.80%	85.40%	80.30%

opinion learning; see Table 1 for examples used in subjectivity classification of weblogs (Huang et al, 2006).

The most typical in opinion mining are binary classification problems, such as learning positive/negative opinions or subjective/objective statements. A few studies have addressed the finer problems of three-class classification (Esuli and Sebastiani, 2006b) and multi-class classification (Wilson et al, 2006). Naïve Bayes, Maximum Entropy, Support Vector Machines and Conditional Random Fields are among the frequently applied algorithms. Also popular but used less often are such algorithms as Fuzzy Clustering, C4.5, Winnow and kNN. In opinion mining, learning algorithms are used in supervised settings (Pang et al, 2002) and in semi-supervised settings (Dredze et al, 2007). In (Dredze et al, 2007), the authors applied Structural Correspondence Learning under supervised and semi-supervised conditions. The algorithm's performance was better on accuracy when the algorithm had access to unlabelled product reviews.

Empirical Comparison of Algorithms in the Learning of Opinions

The high diversity of opinion mining tasks implies the possibility of successful applications for many learning algorithms. If we apply the algorithms on the same data set – to factor out the effect of differences in data characteristics – we can still find text representations that comply better with certain algorithms than with others.

Multinomial Naïve Bayes (NBM), SVM and Maximum Entropy (MaxEntropy) were used to classify movie reviews (Boiy et al, 2007). The authors represent the texts by unigrams, bigrams and adjectives appearing in the data.¹ SVM achieved the best accuracy on texts represented by unigrams, MaxEntropy was the most accurate on the bigram representation, NBM – on texts represented by adjectives. Across the representations, the more linguistic knowledge is held by a representation, the better the performance of NBM. The opposite is true of SVM and MaxEntropy whose performance declines when linguistic knowledge is added. Table 2 presents the empirical results.

Naïve Bayes was used to study the effect of POS tagging on subjectivity classification (Huang 2006). The subject of the study was word subjectivity. To classify a word as subjective or objective, Huang et al (2006) used Naïve Bayes on five data sets, including Yahoo Shopping digital product specifications and users' comments, digital camera reviews and movie reviews. The following representations were built: unigrams (I), bigrams (II), trigrams (III), bigram and trigram Markov chain dependence (IV and V), words with POS tags (VI), POS bigrams and trigrams (VII and VIII), and Markov POS bigram and trigram chain dependence (IX and X). As a result, each data set had ten different representations, five word-based and five POS-based. Micro-averaged F-score estimated the classification results. For each

Table 3. The best Micro F-score (%) of objective/subjective word classification achieved on each data and the representations on which they were obtained; adapted from (Huang et al, 2006)

Data set	F-score	Representation
Movie Review	88.00%	I
Wiebe MPQA	75.30%	II
Hypnia	85.90%	II
Yahoo Shopping	97.60%	IV
Digital Cameras	70.90%	IV

data set, a word-based representation gave the best F-score: II and IV each gave two best representations, I gave one best representation. Thus, the word-based representations dominated the POS-based ones in terms of providing better classification. Table 3 presents the best Micro F-score for each of the data sets.

Some authors take a more traditional approach often used in problems where texts, mostly documents, are classified according to their main topics, that is to say, topic classification. In topic classification problems, texts are frequently represented by statistically designated features, sometimes combined with linguistic analysis. See (Sokolova and Lapalme, 2007) for more on the difference between classification of text topics and subjective texts.

In (Tan and Zhang, 2008), statistical feature selection was used for opinion mining. The authors classified positive/negative opinions of Chinese texts, ChnSentiCorp. They applied four learning algorithms, Naïve Bayes, kNN, SVM and Winnow, to data labeled with part-of-speech tags. Four statistical methods selected words expected to provide better accuracy of opinion classification.

1. Document Frequency Thresholding – words appearing in too few or too many training documents are removed from the text representation. The supporting assumption is that rare and common words are either not informative or not influential.
2. The *CHI* statistics (Galavotti et al, 2000) which assigns each word t its relative statistic $CHI(t) = \max_i CHI(t, c_i)$, where c_i is a classification category,

$$CHI(t, c_i) = \frac{N(AD - BE)^2}{(A + E)(B + D)(A + B)(E + D)}, \quad (5)$$

A is the number of times t and c_i co-occur, B shows how often t appears without c_i , E - the number when c_i occurs without t , and D – neither t nor c_i happen. N is the total number of training documents.

3. Mutual Information (Manning and Schütze, 2003) which associates the word t with the category c_i through $M(t) = \max_i M(t, c_i)$, where

$$MI(t, c_i) = \log \frac{AN}{(A + E)(A + B)} \quad (6)$$

and A, B, E, N are the same as above.

Table 4. The best Micro F-score (%) of positive/negative opinion classification of ChnSentiCorp, adapted from (Tan and Zhang, 2008). Columns list results provided by the feature selection methods, rows – across the classifiers.

Algorithm	DF	CHI	MI	IG
Naïve Bayes	87.00%	89.00%	80.00%	89.00%
SVM	85.00%	89.00%	83.00%	91.00%
KNN	85.00%	84.00%	79.00%	88.00%
Winnow	88.00%	88.00%	81.00%	90.00%

4. Information Gain (Manning and Schütze, 2003) measures the number of bits of information obtained for category prediction by knowing the presence or absence of a word in a document.

$$IG(t) = -\sum_{i=1}^{|C|} P(c_i) \log P(c_i) + P(t) \sum_{i=1}^{|C|} P(c_i | t) \log P(c_i | t) + P(\bar{t}) \sum_{i=1}^{|C|} P(c_i | \bar{t}) \log P(c_i | \bar{t})$$

$P(n)$ denotes the probability that the event n occurs, \bar{t} denotes that the word t does not occur.

The empirical results show that Naïve Bayes benefits when features are either selected with respect to their occurrence in the number of documents – document frequency – or their association with the opinion category – the *CHI* statistics. On the same data, SVM outperforms other algorithms if features are selected with Mutual Information and Information Gain; see Table 4. Note that, although Tables 2 and 4 both present the results of positive/negative opinion classification, a direct comparison is not possible: the results were obtained on different data sets.

Readers familiar with text classification may have noticed that the accuracy and F-score obtained for opinion mining problems are lower than in topic text classification. This could be explained by the ambiguity of the use of opinion-bearing words, which makes the learning task more complex.

NEWS MONITORING

Introduction

The abundance of information – constantly updated – has forced many organizations to switch from manual selection and retrieval of relevant news to automated news monitoring (Zizka et al, 2006). News is sought in multiple sources (including Wikipedia², GoogleNews³ and media email alerts) and in different languages (there are 27 official languages in the European Union alone).

The *Global Public Health Intelligence Network* (GPHIN), successfully deployed by Public Health Agency of Canada, monitors media sources for signals of possible disease outbreaks. GPHIN monitors news on-line in official WHO languages: Arabic, Chinese, English, French, Russian and Spanish (Mawudeku and Blench, 2005). Other important monitoring systems include *HealthMap*, which specializes in monitoring emerging infectious diseases (Freifeld et al, 2008), *NewsExplorer*, which in addition to news extraction automatically generates daily news summaries (Steinberger et al 2006), *MedSys*, a real-time news alert system for medical and health-related topics (Fuart et al, 2007), and *NewsBrief*, whose

Table 5. Samples of article headings extracted for the topic “Nelson Mandela celebrates 90th birthday” from the NewsExplorer website press.jrc.it/NewsExplorer/home/en/latest.html on July 23rd, 2008, at 10:45 am EST.

Language	Three most recent articles covering Nelson Mandela's birthday
English	Boks wish Mandela well on his 90th iol 8:42:00 AM CEST Mandela: a man of all the people independent 1:25:00 AM CEST Mandela's 90th b'day celebrated HindustanTimes 9:06:00 PM CEST
German	Mandela – der Versöhnung wird 90 Heute 7.08 Uhr CEST Ausland: «Hätte gern mehr Zeit mit der Familie verbracht» tagesanzeiger 14.12 Uhr CEST Mit der Kraft der Symbole dna-bilingue 6.20 Uhr CEST
Spanish	Mandela celebra sus 90 años con llamamiento a mantener su legado eluniversal 15H32' CEST Feliz 90 cumpleaños, Nelson Mandela euronews-es 14H12' CEST Nelson Mandela, el gran pacificador de Sudáfrica, cumple hoy 90 años ElComercio 15H17' CEST
French	La presse sud-africaine salut Mandela mais s'inquiète pour son héritage ladepeche 13 h 49 CEST Nelson Mandela, 90 ans, un vieil homme pressé LePoint 08 h 05 CEST Mandela, 90 ans, dénonce inégalités et pauvreté en Afrique du Sud izf 12 h 20 CEST

user-friendly interface clusters live news, displays statistics of the active news categories, provides user-alert services and so on (Best et al, 2005). Table 5 presents a top news topic extracted by *NewsExplorer*. It also lists support articles in English, German, Spanish and French, found by the system.

Apart from overcoming technical implementation difficulties, monitoring systems encounter many NLP problems. They can be classed as unilingual or multilingual: monitoring can be concerned with texts written all in the same language or in different languages. To demonstrate the complexity of unilingual NLP problems, we say that a single task of ascribing news articles to categories – semantic annotation – can be viewed as a hierarchical classification problem. This is the approach taken by (Li et al, 2007) who built an ontology-based IE system which uses the Hieron large-margin learning algorithm for hierarchical classification of news articles.

Cross-document co-reference aimed to find shared hidden topics is another example of the application of ML algorithms (Phan et al, 2008). First, the authors apply a generative model, Latent Dirichlet Allocation, to estimate multinomial observations. They run experiments on Wikipedia and articles from MEDLINE, the largest database of life science references with concentration on biomedical sciences⁴. Next, they train the Maximum Entropy classifier on training sets enhanced with topics found. Disease classification of medical abstracts was improved when the classifier was trained with the topics found in MEDLINE. Improvement also happened for domain classification of Google search snippets when the classifier was trained with the topics found in Wikipedia.

Solving problems of monitoring texts written in different languages includes machine translation and cross-lingual document retrieval or text summarization. Given a query, search for relevant documents written in more than one language usually ensures much better response. Bilingual cross-language retrieval has applied kernel-based learning methods that employ correlation analysis (Vinokourov et al, 2002; Li and Shawe-Taylor, 2006). Cross-language apparatus uses machine translation methods. For example, Markov Chain models (MCM) had been used for query translation in English and Chinese (Cao et al, 2007). Query term co-occurrences and dictionary translations are integrated into direct graphs built by MCM. According to the authors, their procedure propagates monolingual and cross-lingual term similarities among terms in both languages. The method overcomes the limitations imposed by the coverage of the dictionary (such as a weak term disambiguation which leads to many false positive examples). An

Table 6. Classification (%) of focused named entities in a Chinese news corpus. The baseline results are: Precision = 83.09%, Recall = 55.68%, Fscore = 66.68%. The baseline is computed when in-title and most frequent entities are marked as foci.

Dataset	RRM			DT			NB		
	P	R	F	P	R	F	P	R	F
1325 documents	84.70%	78.23%	81.23%	83.83%	74.61%	78.79%	69.14%	89.08%	77.82%

intriguing task is the translation of unknown words. Analogical learning methods can be used to infer such words (Langlais and Patry, 2007).

Named Entity Recognition (NER) is applied to the analysis of information-rich texts such as news reports (Steinberger and Pouliquen, 2007). The handbook chapter “Machine Learning in Natural Language Processing” gives more details on NER. The use of NER methods is important for extracting answers on the five W (who, what, when, where, and why) which essentially summarize monitored news. Using a collection of 1325 documents written in Chinese, (Zhang et al, 2004) compared the performance of three learning algorithms – DT, NB and Robust Risk Minimization Model (RMM) – on the recognition of what they call focused named entities, those which answer the five W questions. The documents were represented by statistical features (including entity frequency, total entity count), significance features (in the title or not, in the first sentence or not, and so on), context information (such as whether neighbours are entities). Table 6 adapted from (Zhang et al, 2004) reports the results.

As we mentioned earlier, deployed monitoring systems navigate news for detection of different topics. However, all systems map the detected events to a location where the event happened. Geographic information detection is a unifying feature of news monitoring. In the remainder of the section we focus on language processing problems in this NLP task. To emphasize its importance, let us note a “competition” of geographic information retrieval systems, GeoCLEF, held annually since 2005.⁵

Problem Description

The detection of geographic information focuses on information retrieval or information resolution tasks. Geographic Information Retrieval (GIR) searches documents with geographically restricted queries. These queries include at least one geographic term such as a place name (Toronto), a suitable adjective (Eastern), a feature (island, lake), a postal code and area code, a descriptive term (state, street). Usually, search for information is based on processing dictionaries of geographic terms and names. This is a keyword-based approach. It gives reliable results if the dictionary entries can be found in searched texts and they cover most, if not all, of mentioned geographic names.

There are several issues typical of Geographic Information Retrieval (Jones and Purves, 2008). They include:

- detection of geographic names; *Tehran* (Iran), *Marseille* (France), *Delhi* (India) may be easier targets than some other places in the same countries, such as *And* (Iran), *This* (France), *She* (India);
- resolution of person-location ambiguity; *Berlin* can denote the capital of Germany and a composer, *Churchill* can refer to a British Prime-Minister and a town, *JFK* is an acronym for a US

president and an international airport;

- matching name variants; *Vienna*, *Vena*, and *Wien* are names the Austrian capital;
- identifying composite names; *Villa Maria* – the name of a villa and an area in Montreal; *Big Apple* – a moniker for *New York* and a reference to a big fruit; *Smith Falls* is a geographical name, whereas *Smith falls* may indicate a person's situation;
- place name disambiguation; *Cambridge*, *London*, *Paris* can be found in Europe and in North America.

Metonymic usage of terms can be ambiguous as well (*Paris* is a metonymy of France and haute-couture; *White House* is an internationally recognized metonymy for the US central administration as well as a metonymy for the Russian parliament). More examples of ambiguous geographic names are listed by Pouliquen *et al.* (2006) who present empirical evidence for Geographic Information Extraction challenges in English, French, and German.

The performance of GIR systems relies on specialized resources of geographic names, such as the Getty Thesaurus of Geographic Names⁶ and the World Gazetteer⁷. Recall that a gazetteer is a dictionary of geographic names used in a conjunction with a map. Most of these resources provide global geographic information which may not be comprehensive in covering local data. Access to complete local information can overcome some of the problems. Anecdotal evidence shows the necessity of such access even for the best available software. For example, LingPipe⁸ is arguably the extraction engine that most GIR systems currently use. It has access to global, but not local, geographic information. We tested the system on the recognition of the Canadian capital *Ottawa* and a university/resort city *Kingston* located in the province of Ontario, Canada. It recognized *Ottawa* as a geographical entity, whereas *Kingston* was labeled as an organization (obtained on March 7, 2008):

```
<ENAMEX TYPE="LOCATION">Ottawa</ENAMEX> gets a record snowstorm.  
<ENAMEX TYPE="ORGANIZATION">Kingston</ENAMEX> gets a record snow-  
storm.
```

In this case, adding information about the province of Ontario allows correct name recognition.

To resolve more difficult cases, for example disambiguate *Smith Falls*, the systems turn to the linguistic analysis of topics or documents. They work with both rule-based and machine learning methods. Toponym and metonym resolution is often performed by probabilistic methods using empirical evidence – local contextual information, population information, global contextual information, and so on. To find more about the toponym and metonym resolution methods see (Leiden 2008). Although GIR systems can perform information resolution tasks, Geographic Information Resolution (GIRE) systems are built for large-scale applications.

GIR and GIRE are domain- and task-focused. That is why they are often custom-built to serve specific applications. In addition to the systems listed above, we list two medical surveillance systems. Both systems detect geographic information in texts.

1. The *BioCaster Global Health Monitor*⁹ has been developed for disease rumour surveillance. The currently deployed version focusses on communicable disease surveillance from the Internet news. It is based on a text-mining system with knowledge to recognize important concepts in the domain (Collier *et al.*, 2007). The core text-mining module applies machine learning to a multilingual

collection of news articles. The system monitors news delivered in official languages of the Asia-Pacific region. Its geographic taxonomy is based on a hierarchy of regions, among which Continent is the highest level (Doan et al, 2008).

2. *PULS*¹⁰ – a pattern-based, language understanding and machine learning system – is reported to be in advanced development stages. The system is being developed to extract factual information from epidemiological reports (Steinberger et al, 2008; Yangarber et al, 2008). It uses GIR and GIRE. Its natural language analysis apparatus complements the apparatus of *MedSys*, which is specifically designed for medical news monitoring. In *PULS*, the knowledge bases are populated by weakly-supervised machine learning methods which reduce the amount of manual labour.

Initially, GIR and GIRE systems relied on the use of dictionaries and rule-based search. In recent years, the systems' text analysis part increasingly involves NLP reinforced by ML methods.

Learning Algorithms in Geographic Information Detection

The performance of ML methods in geographic information detection can be compared from several points of view. In this study, we want to emphasize the effect of training data on the algorithm performance.

The algorithms must have access to annotated training data which accurately and, ideally, completely represent the tasks which the algorithms will be solving. The effect of the proper training data could be seen on the following example: the performance of two off-the shelf systems, LingPipe and OpenNLP NERC¹¹, was compared on data that combined North American and European information (Stokes et al, 2008). The goal was to identify place names (e.g., *Georgia*, *Waterloo*) in the GeoCLEF competition data. The data consists of the *Los Angeles Times* and the *Glasgow Herald* collections. LingPipe (the learning component – HMM) and OpenNLP NREC (the learning component – Maximum Entropy) were trained mostly on North American international news sources and did not do well. Over-fitted on specific geographic information, they performed worse than G-lookup which implemented a rule-based search based on the Getty Thesaurus of Geographic Names: LingPipe's F-score = 55.44%, OpenNLP NERC's F-score = 53.78%, G-Lookup's F-score = 59.74%. This case illustrates the fact that, although the application of suitably chosen ML techniques to the detection of geographic information is a very promising avenue, in certain circumstances there is no need to deploy elaborate methods.

A more detailed study (Surdeanu et al, 2005) showed that sometimes adding gazetteer information may actually worsen the identification of location names. The authors applied SVM to named entity recognition and classification at the Switchboard corpus¹², records of spontaneous telephone conversation, totalling 240 hours and 3 million words. A US location gazetteer¹³ and a world location GeoNS gazetteer¹⁴ were used, as well as the databases of first and last person names from the US Census¹⁵. Binary classification problems were solved to classify entities of several classes including Person names, Location names, and Organization names. Words were represented by several features sets. The classification results on each representation were compared. We list F-score results in Table 7, where *lexical features* mean lexemes, affixes (prefixes and suffixes) and non-alphanumeric features, *format features* mean uppercase letters, dots, digits, *gazetteer features* mean the presence or absence in a known gazetteer or a database. POS features gave part-of-speech information. The best F-score for location names were obtained on a combination of lexical, format and POS features.

The same types of named entities were better classified in terms of F-score when ML methods were

Table 7. F-score (%) of location, organization, and person named entities classification. The entities appear in the Switchboard corpus. One-versus-all classification was performed by SVM with kernels of degree 2.

Named entity category	Lexical features	Lexical and Format features	Lexical, format, and POS features	Lexical, format, POS, and gazetteer features
Location	81.13%	83.50%	83.92%	83.11%
Organization	62.63%	63.33%	64.21%	64.83%
Person	62.48%	74.27%	72.73%	77.72%

applied to Reuters documents (the CoNLL 2003 training corpus). (Florian et al, 2003) showed that F-score of location name classification can be as high as 91.15% (precision = 90.59%, recall = 91.73%) if words are represented by their lemmas, POS tags, affixes and format tags. Applied learning method was a partial credit combination of Robust Risk Minimization, Maximum Entropy, Transformation-based Learning, and HMM. We can partially attribute the higher F-score to a better quality (grammar, structure, spelling, absence of noise) of Reuters document texts when compared with the Switchboard corpus.

To put Named Entity Recognition into perspective, we emphasize that this is one part of a larger NLP task of geographic information detection. The detection methods include lexical and semantic processing and multi-modal data analysis. Challenges posed by these problems make the geographic information detection and news monitoring systems development more interesting tasks.

PRIVACY PROTECTION

Introduction

Electronic storage of vast numbers of documents has become routine for many organizations. Data contained in these documents are sought after by researchers, businesses and government agencies. The organizations that collect and keep highly sensitive personal information are bound by law to protect the information from unintentional disclosure when giving it to third parties. Although Europe and North America differ in details of information protection standards, it is a common requirement that personal identifiers be eliminated before data transfer (Elger and Caplan, 2006). Personal identity protection, referred to as *de-identification* and *anonymization*, is at the forefront of research into the joint contribution of NLP and ML to the solving of language classification problems. ML has been applied to the recognition of person identifiers, such as names and signatures in email (Carvalho and Cohen, 2004; Minkov et al, 2005) and in any documents in electronic format (Aura 2006). Most de-identification and anonymization methods have been specifically designed for English. A language-independent de-identification algorithm has been implemented in the anonymization system which works with legal and court documents (Plamondon 2004). The system is based on the language engineering platform GATE¹⁶. De-identification applies to documents written in English, French, German, and Spanish. The procedure finds person names and replaces them with initials (John Doe → J.D.). It finds dates of person-identifying events, e.g., birth and death, and replaces them with partial dates (born on February 28, 2000 → born [...], 2000).

Research on text de-identification is most intensive in health care. In medical organizations, electronic

means have become the prime medium for storage and exchange of information, including patient records (Brox Røst et al, 2006). These text files, often in free form, necessarily contain identifiable patient information and private health information. Henceforth, we will concentrate on the application of ML methods to de-identification of medical documents.

Problem Description

Personal health information (PHI) in medical documents consists of *identity information* (II) such as person names, street addresses, phone numbers or IDs, and *health information* (HI) which includes physical symptoms and conditions, disease diagnoses, health care providers and so on. Many documents have parts of unstructured text, written in a free format. Free-form text allows a large diversity in style and spelling; this introduces a considerable amount of language noise. Large volumes of noisy data require robust and efficient processing techniques. The following excerpt, adapted from (Uzuner et al, 2007), shows an example of a patient's discharge summary. In the excerpt, the actual name of the patient and the unit number were changed to protect his personal information:

Discharge Summary Name:

Sloan, Charles E.

Unit Number: 358-51-76

Principal Diagnosis:

Carcinoma of the colon

Associated Diagnosis:

Urinary tract infection, and cirrhosis of the liver

History of Present Illness:

The patient is an 80-year-old male, who had a history of colon cancer in the past, resected approximately ten years prior to admission, history of heavy alcohol use, who presented with a two week history of poor PO intake...

Several ML-based systems have been developed to prevent leaking identifying information in medical texts. A direct comparison of the systems' performance on several data sets would help illustrate the benefits of each system, but benchmark data sets for privacy protection are yet to be built. Most of the methods employed first find identifiable personal information with further de-identification of records. Next, they apply such principles and methods that the remaining information cannot be used, alone or in combination with other reasonably available information, to identify an individual who is the subject of the information. See, for example, (Wellner et al, 2007).

A few methods are concerned with anonymization: the removal of person-related information that could be used for tracking the actual person from documents. The anonymization process, considering all available information about a person, is more complex than de-identification which works with a restricted number of person characteristics (Wojcik et al, 2007).

Name Entity Recognition (NER) was the first technique chosen to build de-identification systems (Sweeney 1996). Systems based only on NER, however, depend to a high degree on the structure of texts. Systems that apply both NER and ML have proved more flexible and rapidly gained popularity among researchers and practitioners (Uzuner et al, 2007).

De-identification systems currently aim for high recall of PHI identification. In the longer run, how-

ever, a challenge that most of the algorithms try to meet is the avoidance of false positives, including de-identification of unnecessary information. For example, *Smith* must be de-identified if it is a last name, but can be left unchanged if it is a part of *Smith Falls*, a town in Canada.

Empirical Comparison of Learning Algorithms in Person De-Identification

Several systems that assist in preventing leaks of textual private information have been deployed (Uzuner et al, 2007). The learning algorithms commonly used to identify personal information are Support Vector Machines, Conditional Random Fields, and rule-based learning.

De-identification tasks pose interesting text analysis tasks. Often applied to hospital records, they partially work with doctor and nurse notes, often hastily written with little or no respect for grammar, punctuations, proper capitalization, etc. For example, *hepatitis B* can be short-handed as *hep.b*. It is natural to expect the applied methods to be robust and noise-resistant. On the other hand, in de-identification of hospital discharge summaries, especially gathered at the same hospital, texts have an expected flow. A summary might go as follows: a general heading (file number, medical provider's name, date and time, final diagnosis), a specific heading (patient's name, department ID, admission and discharge dates), diagnosis and the history of the illness. Only the latter is a free-form text. Thus, accommodating the text structure into learning paradigm can help with improvement of results.

Still, the texts can be represented by a relatively restricted number of text features, due to standardized, albeit non-standard, vocabulary. In such cases, structural algorithms CRF and Hidden Markov Models (HMMs) and DT sometimes outperform SVM (Aramaki and Miyo, 2006; Hara 2006; Szarvas and Busa-Fekete, 2006; Wellner et al, 2007). These systems apply *text chunking* that divides text into syntactically connected non-overlapping segments (for more details, refer to the chapter “Machine Learning in Natural Language Processing”). CRF and HMM build sequential models to learn text-dependent features (e.g., the beginning and end of a chunk or sentence) and external features (e.g., names, locations, dates) important to further identification of personal and health information.

A comparison of classification performance of the algorithms in de-identification systems appears in (Uzuner et al, 2007). The task was a binary classification problem: determine whether a word is PHI. The data set was built from 889 records of authentic patient discharge summaries gathered at the same hospital. The algorithms used training/test split for the model selection: 669 records were used to train, the remaining 220 – to test the systems.

In the data, all authentic personal information indicators were replaced with surrogates generated by permutation of characters, letters for personal and geographic names and organizations and digits for dates, IDs, age and phone numbers. According to the data contributors, the applied text engineering reduced the ambiguity for geographic names, phone numbers, and dates, thus making it easier for ML algorithms to classify words. The number of ambiguous geographic names fell from 43.9% in the original corpus to 18.1% and 14.3% in the training and test sets respectively. Although the partial data disambiguation created a simplified language environment, it provided the first chance for direct comparison of performance of several de-identification systems. Five systems used SVM, three systems each – CRF and DT. In terms of correct classification of PHI labels, all the tested CRF- and DT-based systems performed significantly better than SVM-based systems (a randomization technique, $\alpha=0.1$).

When the number of text features increases substantially, an SVM-based system achieves highly accurate de-identification. In (Uzuner et al, 2008), each word was classified as PHI or not PHI. Several systems were run on two sets of manually edited summaries (containing 48 and 889 summaries respec-

Table 8. Classification (%) of PHI words in discharge summaries. The measures are calculated with respect to classification of PHI words; adapted from (Uzuner 2008).

Systems	Results on randomized corpus			Results on authentic corpus		
	F-score	Precision	Recall	F-score	Precision	Recall
SVM-based (Stat De-id)	97.63%	98.34%	96.92%	96.82%	98.46%	95.24%
CRF-based	81.55%	81.94%	81.17%	87.83%	91.16%	84.75%
H+D-based	77.82%	93.67%	66.56%	84.92%	82.67%	87.30%

tively) and on data extracted from 90 authentic summaries. The summaries were gathered in hospitals in two countries. Consequently, the records were structurally less consistent than the records discussed in (Uzuner et al, 2007). Perhaps this diversity prompted the researchers to seek additional information about the texts.

The texts were tagged by the Link Grammar Parser (Grinberg et al, 1995) and the Brill tagger (Brill 1995) with links between words and part-of-speech tags. Next, the feature set was expanded by terms extracted from medical dictionaries and by lexical and orthographic features. Thus, a word was represented by thousands of features for binary word classification. The results of the SVM-based system, Stat De-id, were compared with those of a CRF-based system and a heuristic + dictionary system (H+D in Table 8). Partial comparison was made with an HMM-based model (the systems were trained on different data sets) and a probabilistic-reasoning model (the system performance was compared only on a few PHI categories). The best models were chosen by tenfold cross-validation. Table 8, adapted from (Uzuner et al, 2008), presents F-score, Precision and Recall calculated with respect to the classification of PHI words. Note that Stat De-id's performance deteriorates on authentic non-engineered summaries, whereas the performance of CRF- and H+D-based systems improves if compared with their results on preliminary engineered summaries. For both systems the improvement is close to, but does not reach, statistical significance according to paired *t*-test: P=0.0592 for the CRF-based system and P=0.603 for the H+D system.

Although the reported performance results are high, it would be practical to know how the systems perform without access to customized information, e.g., customized dictionaries, and on documents with large volumes of free texts. In (Neamatullah et al, 2008), the authors apply Stat De-id to de-identify nurse notes. The notes contain larger volumes of free form texts than discharge summaries. The authors first use the tool with access to customized dictionaries and then without it. Table 9, adapted from (Neamatullah et al, 2008), reports the results. On nurse notes, Stat De-id has a sharp, almost 22%, drop in Precision, even when customized dictionaries are available. Without customized dictionaries, both Precision and Recall deteriorate, although not statistically significantly (paired *t*-test, P=0.1332).

In the end, we can say that such results of de-identification system performance support the “no free

Table 9. Classification (%) of PHI words in nurse notes. The measures are calculated with respect to PHI words.

System	Results with customized dictionaries			Results without customized dictionaries		
	Fscore	Precision	Recall	Fscore	Precision	Recall
Stat De-id	84.40%	74.90%	96.70%	77.40%	72.30%	83.40%

lunch” theorem (Wolpert and Macready, 2005): no algorithm can outperform all others on all problems. We have shown that different algorithms may solve the same binary classification problem better in different data settings. The choice of algorithm may relate to data characteristics, features and representation.

CONCLUDING REMARKS

In this chapter we have concentrated on ML applications to NLP problems that involve mega-texts. Language studies in this direction began in earnest when the Internet and computer-based technologies became a prime medium for text storage and exchange. We chose the applications that are not only most popular but also important both socially and economically: opinion mining, news monitoring and privacy protection. The discussion of the three applications has barely scratched the surface of the use of ML methods, because ML is instrumental in nearly all NLP work.

The volume of mega-text data – immeasurably far beyond anyone’s manual processing capacity – means that the ML-NLP connection is a natural match that benefits both fields. ML researchers get to work with unusually copious, varied and challenging data. NLP researchers get to use powerful tools that can bring out new insights into the problem solutions. However, the News Monitoring results we showed tell us that it would be detrimental to completely ignore simple lookup. On the other hand, Opinion Mining studies we referred to showed the importance of elaborate semantic processing.

The Privacy Protection results we cited demonstrate that CRF and HMM perform equally well as, or even better than, DT, NB, kNN and SVM, the ML algorithms still by far the most used in the NLP community. The no-clear-winner empirical results make it difficult to find a perfect fit (one well-known ML technique per one NLP task). It is possible that algorithms with inferior performance had not been designed, or fine-tuned, for the tasks they were applied to – like an algorithm trained on North American data sets but applied to European data.

Some mega-text language processing problems pose additional questions for ML applications. For example, in de-identification of personal information decision it is important for feature selection and subsequent text representation choices to know whether texts can be treated as structured and well-edited documents. Knowing that a text is a conversation record (rather than content-governed, such as a seminar presentation) can be important for deciding between static and dynamic learning techniques. Text length – the number of words – is also an important factor. For example, shorter texts usually carry less information. If you work with short texts, try to use data representations and algorithms that extract as much information from texts as possible; for example, N -gram-based classifiers built with characters could be as accurate as NB and SVM (Bobicev and Sokolova, 2008).

Our presentation focussed on a detailed coverage of ML algorithms applied to three mega-text language processing topics. There are many other topics that can be considered within the mega-text agenda. For example, Wikipedia is a unique phenomenon that relies on open contribution and editing of texts. This mega-text collection, currently perhaps the largest single knowledge repository, became a major subject of NLP studies (Gabrilovich and Markovitch, 2007; Egozi et al, 2008; Nastase and Strube, 2008). Nonetheless, we feel that we have given the reader enough information to find her way around the field, if only by naming the necessary search terms. For example, if the reader is interested in domain adaptation of text classification methods, then, searching for the ML application components, she could find out that hold-out sets, not cross-validation technique, would be a model selection method of choice.

ABBREVIATIONS

CRF: Conditional Random Fields
DT: Decision Trees
GIR: Geographic Information Retrieval
GIRE: Geographic Information Resolution
HI: Health Information
HMM: Hidden Markov Model
IE: Information Extraction
II: Identity Information
kNN: k-Nearest Neighbor
MCM: Markov Chain model
NB: Naïve Bayes
NER: Name Entity Recognition
PHI: Personal Health Information
POS: Part-of-Speech
SVM: Support Vector Machines

ACKNOWLEDGMENT

We thank Khaled El Emam for helpful comments on the chapter. This work is partially supported by the Ontario Centres of Excellence and the Natural Sciences and Engineering Research Council of Canada.

REFERENCES

- Andreevskaia, A., & Bergler, S. (2006). *Mining WordNet for fuzzy sentiment: Sentiment tag extraction from WordNet glosses*. Paper presented at the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL'06).
- Aramaki, E., & Miyo, K. (2006). *Automatic deidentification by using sentence features and label consistency*. Paper presented at the i2b2 Workshop on Challenges in Natural Language Processing for Clinical Data.
- Aura, T., Kuhn, T., & Roe, M. (2006). *Scanning electronic documents for personally identifiable information*. Paper presented at the 5th ACM workshop on Privacy in electronic society.
- Benamara, F., Cesarano, C., Picariello, A., Reforgiato, D., & Subrahmanian, V. (2007). *Sentiment analysis: Adjectives and adverbs are better than the adjectives alone*. Paper presented at the International Conference on Weblogs and Social Media (ICWSM'2007).

- Best, C., van der Goot, E., Blackler, K., Garcia, T., Horby, D., Steinberger, R., & Pouliquen, B. (2005). Mapping world events. In P. Van Oosterom, S. Zlatanova, & E. Fendel (Eds.), *Geo-information for disaster management* (pp. 683-696). Berling, Germany: Springer.
- Bobicev, V., & Sokolova, M. (2008). *An effective and robust method for short text classification*. Paper presented at the Twenty-Third Conference on Artificial Intelligence (AAAI 2008).
- Boiy, E., Hens, P., Deschacht, K., & Moens, M.-F. (2007). *Automatic sentiment analysis in on-line text*. Paper presented at the Conference on Electronic Publishing (ELPUB 2007).
- Breck, E., Choi, Y., & Cardie, C. (2007). *Identifying expressions of opinion in context*. Paper presented at the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-2007).
- Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4), 543–565.
- Brox Røst, T., Nytrø, Ø., & Grimsmo, A. (2006). *Classifying encounter notes in the primary care patient record*. Paper presented at the ECAI'06 3rd International Workshop on Text-based Information Retrieval.
- Cao, G., Gao, J., Nie, J.-Y., & Bai, J. (2007). *Extending query translation to cross-language query expansion with Markov chain models*. Paper presented at the Conference on Information and Knowledge Management (CIKM 2007).
- Carvalho, V., & Cohen, W. (2004). *Learning to extract signature and reply lines from email*. Paper presented at the First Conference on Email and Anti-Spam.
- Charniak, E. (2000). *A maximum entropy inspired parser*. Paper presented at the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2000).
- Collier, N., Kawazoe, A., & Jin, L. (2007). A multilingual ontology for infectious disease outbreak surveillance: Rationale, design and challenges. *Journal of Language Resources and Evaluation*, 40(3-4), 405–413. doi:10.1007/s10579-007-9019-7
- Crystal, D. (2006). *Language and the Internet*. Cambridge, UK: Cambridge University Press.
- Doan, S., Hung-Ngo, Q., Kawazoe, A., & Collier, N. (2008). *Global health monitor - a Web-based system for detecting and mapping infectious diseases*. Paper presented at the International Joint Conference on Natural Language Processing (IJCNLP - 2008).
- Dredze, M., Blitzer, J., & Pereira, F. (2007). *Biographies, Bollywood, boom-boxes, and blenders: Domain adaptation for sentiment classification*. Paper presented at the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007).
- Egozi, O., Gabrilovich, E., & Markovitch, S. (2008). *Concept-based feature generation and selection for information retrieval*. Paper presented at the Twenty-Third Conference on Artificial Intelligence (AAAI 2008).
- Elger, B., & Caplan, A. (2006). Consent and anonymization in research involving biobanks. *European Molecular Biology Organization reports*, 7(7), 661-666.

- Esuli, A., & Sebastiani, F. (2006a). *Determining term subjectivity and term orientation for opinion mining*. Paper presented at the European Chapter of the Association for Computational Linguistics (EACL'07).
- Esuli, A., & Sebastiani, F. (2006b). *SENТИWORDNET: A publicly available lexical resource for opinion mining*. Paper presented at the 5th Conference on Language Resources and Evaluation (LREC 2006).
- Esuli, A., & Sebastiani, F. (2007). *Random-walk models of term semantics: An application to opinion-related properties*. Paper presented at the 3rd Language and Technology Conference (LTC 2003).
- Fellbaum, C. (1998). *WordNet: An electronic lexical database* Cambridge, MA: The MIT Press.
- Florian, R., Ittycheriah, A., Jing, H., & Zhang, T. (2003). *Named entity recognition through classifier combination*. Paper presented at the Seventh Conference on Natural Language Learning (CoNLL 2003).
- Freifeld, C., Mandl, K., Reis, B., & Brownstein, J. (2008). HealthMap: Global infectious disease monitoring through automated classification and visualization of Internet media reports. *Journal of the American Medical Informatics Association*, 15, 150–157. doi:10.1197/jamia.M2544
- Fuart, F., Horby, D., & Best, C. (2007). *Disease outbreak surveillance through the Internet - the MedISys project*. Paper presented at the European Federation for Medical Informatics Special Topic Conference.
- Gabrilovich, E., & Markovitch, S. (2007). *Computing semantic relatedness using Wikipedia-based explicit semantic analysis*. Paper presented at the 20th International Joint Conference on Artificial Intelligence (IJCAI'07).
- Galavotti, L., Sebastiani, F., & Simi, M. (2000). *Feature selection and negative evidence in automated text categorization*. Paper presented at the 4th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2000).
- Grinberg, D., Lafferty, J., & Sleator, D. (1995). *A robust parsing algorithm for link grammars*. Paper presented at the Fourth International Workshop on Parsing Technologies.
- Hara, K. (2006). *Applying a SVM based chunker and a text classifier to the Deid challenge*. Paper presented at the i2b2 Workshop on Challenges in Natural Language Processing for Clinical Data.
- Hu, M., & Liu, B. (2004). *Mining and summarizing customer reviews*. Paper presented at the International Conference on Knowledge Discovery and Data Mining (KDD'04).
- Huang, S., Sun, J.-T., Wang, X., Zeng, H.-J., & Chen, Z. (2006). *Subjectivity categorization of Weblog with part-of-speech based smoothing*. Paper presented at the Sixth International Conference on Data Mining (ICDM'06).
- Jindal, N., & Liu, B. (2008). *Opinion spam and analysis*. Paper presented at the International Conference on Web Search and Web Data Mining.
- Jones, C., & Purves, R. (2008). Geographical information retrieval. *International Journal of Geographical Information Science*, 22(3), 219–228. doi:10.1080/13658810701626343

- Kim, S.-M., & Hovy, E. (2004). *Determining the sentiment of opinions*. Paper presented at the 20th International Conference on Computational Linguistics (COLING-04).
- Kim, S.-M., & Hovy, E. (2007). *CRYSTAL: Analyzing predictive opinions on the Web*. Paper presented at the Conference on Empirical Methods in Natural Language Processing (EMNLP-CoNLL 2007).
- Kudo, T., & Matsumoto, Y. (2003). *Fast methods for kernel-based text analysis*. Paper presented at the 41st Annual Meeting on Association for Computational Linguistics.
- Langlais, P., & Patry, A. (2007). *Translating unknown words by analogical learning*. Paper presented at the Empirical Methods in Natural Language Processing (EMNLP-CoNLL 2007).
- Leiden, J. (2008). *Toponym resolution in text: Annotation, evaluation and applications of spatial grounding of place names*. Retrieved from <http://www.dissertation.com>
- Li, Y., Bontcheva, K., & Cunningham, H. (2007). *Hierarchical, perceptron-like learning for ontology based information extraction*. Paper presented at the Proceedings of the 16th International World Wide Web Conference (WWW2007).
- Li, Y., & Shawe-Taylor, J. (2006). Using KCCA for Japanese---English cross-language information retrieval and document classification. *Journal of Intelligent Information Systems*, 27(2), 117–133. doi:10.1007/s10844-006-1627-y
- Liu, B. (2006). *Web data mining - exploring hyperlinks, contents and usage data*. Berlin, Germany: Springer.
- Manning, C., & Schütze, H. (2003). *Foundations of statistical natural language processing* (6th ed.). Cambridge, MA: The MIT Press.
- Marcus, M., Santorini, B., & Marcinkiewicz, M. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2), 313–330.
- Mawudeku, A., & Blench, M. (2005). *Global public health intelligence network (GPHIN), invited talk*. Paper presented at the The Tenth Machine Translation Summit.
- Minkov, E., Wang, R., & Wang, W. (2005). *Extracting personal names from email: Applying named entity recognition to informal text*. Paper presented at the Empirical Methods in Natural Language Processing.
- Mladenic, D. (2002). *Automatic word lemmatization*. Paper presented at the 5th International Multi-Conference: Information Society (IS 2002).
- Nastase, V., & Strube, M. (2008). *Decoding Wikipedia categories for knowledge acquisition*. Paper presented at the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI 2008).
- Neamatullah, I., Douglass, M., Lehman, L., Reisner, A., Villarroel, M., Long, W., Szolovits, P., Moody, G., Mark, R., & Clifford, G. (2008). Automated de-identification of free-text medical records. *Medical Informatics and Decision Making*, 8(32).
- Pang, B., Lee, L., & Vaithyanathan, S. (2002). *Thumbs up? Sentiment classification using machine learning techniques*. Paper presented at the Empirical Methods of Natural Language Processing (EMNLP'02).

- Pennebaker, J., Francis, M., & Booth, R. (2001). *Linguistic inquiry and word count* (2nd ed.). New York: Psychology Press.
- Phan, X.-H., Nguyen, L. M., & Horiguchi, S. (2008). *Learning to classify short and sparse text & Web with hidden topics from large-scale data collections*. Paper presented at the 17th International Conference on World Wide Web (WWW 2008).
- Plamondon, L., Lapalme, G., & Pelletier, F. (2004). *Anonymisation de décisions de justice*. Paper presented at the XIe Conférence sur le Traitement Automatique des Langues Naturelles (TALN 2004).
- Pouliquen, B., Kimler, M., Steinberger, R., Ignat, C., Oellinger, T., Blackler, K., et al. (2006). *Geocoding multilingual texts: Recognition, disambiguation and visualisation*. Paper presented at the 5th International Conference on Language Resources and Evaluation (LREC'2006).
- Sokolova, M., & Lapalme, G. (2007). *Performance measures in classification of human communication*. Paper presented at the 20th Canadian Conference on Artificial Intelligence (AI'2007).
- Steinberger, R., Fuart, F., van der Goot, E., Best, C., von Etter, P., & Yangarber, R. (2008). Text mining from the Web for medical intelligence. In D. Perrotta, J. Piskorski, F. Soulié-Fogelman, & R. Steinberger (Eds.), *Mining massive data sets for security*. Amsterdam: OIS Press.
- Steinberger, R., & Pouliquen, B. (2007). Cross-lingual named entity recognition. *Linguisticae Investigationes*, 30(1), 135–162.
- Steinberger, R., Pouliquen, B., Widiger, A., Ignat, C., Ejavec, T., Tufis, D., & Varga, D. (2006). *The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages*. Paper presented at the 5th International Conference on Language Resources and Evaluation (LREC 2006).
- Stokes, N., Li, Y., Moffat, A., & Rong, J. (2008). An empirical study of the effects of NLP components on geographic IR performance. *International Journal of Geographical Information Science*, 22(3), 247–264. doi:10.1080/13658810701626210
- Strapparava, C., Valitutti, A., & Stock, O. (2006). *The affective weight of the lexicon*. Paper presented at the 5th International Conference on Language Resources and Evaluation (LREC 2006).
- Surdeanu, M., Turmo, J., & Comelles, E. (2005). *Named entity recognition from spontaneous open-domain speech*. Paper presented at the Interspeech 2005–Eurospeech.
- Sweeney, L. (1996). Replacing personally-identifying information in medical records, the scrub system. *Journal of the American Medical Informatics Association*, 333–337.
- Szarvas, G., & Busa-Fekete, R. (2006). *State-of-the-art anonymization of medical records using an iterative machine learning framework*. Paper presented at the i2b2 Workshop on Challenges in Natural Language Processing for Clinical Data.
- Tan, Z., & Zhang, J. (2008). An empirical study of sentiment analysis for Chinese documents. *Expert Systems with Applications*, 34(4), 2622–2629. doi:10.1016/j.eswa.2007.05.028

- Thomas, M., Pang, B., & Lee, L. (2006). *Get out the vote: Determining support or opposition from congressional floor-debate transcripts*. Paper presented at the Empirical Methods in Natural Language Processing (EMNLP 2006).
- Turney, P., & Littman, M. (2003). Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4), 315–346. doi:10.1145/944012.944013
- Uzuner, O., Luo, Y., & Szolovits, P. (2007). Evaluating the state-of-the-art in automatic de-indentification. *Journal of the American Medical Informatics Association*, 14, 550–563. doi:10.1197/jamia.M2444
- Uzuner, O., Sibanda, T., Luo, Y., & Szolovits, P. (2008). A de-identifier for medical discharge summaries. *Journal of Artificial Intelligence in Medicine*, 42, 13–35. doi:10.1016/j.artmed.2007.10.001
- Vinokourov, A., Shawe-Taylor, J., & Cristianini, N. (2002). *Inferring a semantic representation of text via cross-language correlation analysis*. Paper presented at the Neural Information Processing Systems (NIPS 2002).
- Wellner, B., Huyck, M., & Mardis, S. (2007). Rapidly retargetable approaches to de-identification in medical records. *Journal of the American Medical Informatics Association*, 14, 564–573. doi:10.1197/jamia.M2435
- Wiebe, J., & Riloff, E. (2005). *Creating subjective and objective sentence classifiers from unannotated texts*. Paper presented at the Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2005).
- Wilson, T., Wiebe, J., & Hwa, R. (2006). Recognizing strong and weak opinion clauses. *Computational Intelligence*, 22(2), 73–99. doi:10.1111/j.1467-8640.2006.00275.x
- Wojcik, R., Hauenstein, L., Sniegoski, C., & Holtry, R. (2007). Obtaining the data. In J. Lombardo & D. Buckeridge (Eds.), *Disease surveillance* (pp. 91–142). New York: Wiley.
- Wolpert, D., & Macready, W. (2005). Coevolutionary free lunches. *IEEE Transactions on Evolutionary Computation*, 9(6), 721–735. doi:10.1109/TEVC.2005.856205
- Yangarber, R., von Etter, P., & Steinberger, R. (2008). *Content collection and analysis in the domain of epidemiology*. Paper presented at the International Workshop on Describing Medical Web Resources, held in conjunction with the 21st International Congress of the European Federation for Medical Informatics (MIE 2008).
- Zhang, L., Pan, Y., & Zhang, T. (2004). *Focused named entity recognition using machine learning*. Paper presented at the 27th Annual International ACM SIGIR Conference (SIGIR 2004).
- Zhang, T., Damerau, F., & Johnson, D. (2002). Text chunking based on a generalization of Winnow. *Journal of Machine Learning Research*, 2, 615–637. doi:10.1162/153244302320884560
- Zizka, J., Hroza, J., Pouliquen, B., Ignat, C., & Steinberger, R. (2006). *the selection of electronic text documents supported by only positive examples*. Paper presented at the 8es Journees internationales d'Analyse statistique des Donnees Textuelles (JADT 2006).

KEY TERMS AND DEFINITIONS

Natural Language Processing: theory, design and implementation of systems for the analysis, understanding and generation of written or spoken language.

Mega-Text: large volumes of Web-based, computer-mediated, and other electronic-format texts.

Mega-Text Language Processing: Natural Language Processing applied to large volumes of Web-based, computer-mediated, and other electronic-format texts.

Opinion Mining: an automatic and semi-automatic search for expressed opinions in texts.

News Monitoring: automated tracking of online news.

Privacy Protection in Texts: protection of personal information that could reveal a person's identity.

Text Classification: automatic assigning a text with a tag, chosen from a set of tags.

ENDNOTES

¹ N -gram models are computed as $P(w_k | w_1^{k-1}) \approx P(w_k | w_{k-N}^{k-1})$, where $P(w_k | w_{k-N}^{k-1})$ is the probability of the word w_k appearing after the sequence $w_{k-N} \dots w_{k-1}$.

² <http://www.wikipedia.org/>

³ <http://news.google.ca/nwshp?hl=en&tab=wn>

⁴ http://www.nlm.nih.gov/databases/databases_medline.html

⁵ <http://ir.shef.ac.uk/geoclef/>

⁶ http://www.getty.edu/research/conducting_research/vocabularies/tgn/

⁷ <http://world-gazetteer.com/>

⁸ <http://alias-i.com/lingpipe/>

⁹ <http://biocaster.nii.ac.jp/>

¹⁰ <http://www.cs.helsinki.fi/group/ohtu/s-2008/aiheet/PULS.html>

¹¹ <http://opennlp.sourceforge.net/>

¹² http://www.ldc.upenn.edu/Catalog/readme_files/switchboard.readme.html

¹³ <http://geonames.usgs.gov>

¹⁴ <http://earth-info.nga.mil>

¹⁵ <http://www.census.gov>

¹⁶ <http://gate.ac.uk/>

Chapter 16

FOL Learning for Knowledge Discovery in Documents

Stefano Ferilli

Università degli Studi di Bari, Italy

Floriana Esposito

Università degli Studi di Bari, Italy

Marenglen Biba

Università degli Studi di Bari, Italy

Teresa M.A. Basile

Università degli Studi di Bari, Italy

Nicola Di Mauro

Università degli Studi di Bari, Italy

ABSTRACT

This chapter proposes the application of machine learning techniques, based on first-order logic as a representation language, to the real-world application domain of document processing. First, the tasks and problems involved in document processing are presented, along with the prototypical system DOMINUS and its architecture, whose components are aimed at facing these issues. Then, a closer look is provided for the learning component of the system, and the two sub-systems that are in charge of performing supervised and unsupervised learning as a support to the system performance. Finally, some experiments are reported that assess the quality of the learning performance. This is intended to prove to researchers and practitioners of the field that first-order logic learning can be a viable solution to tackle the domain complexity, and to solve problems such as incremental evolution of the document repository.

INTRODUCTION

After some years in which it was seen as an area of interest only for research, Machine Learning (ML for short) techniques and systems have started to gain progressive credit in the everyday Computer Sci-

DOI: 10.4018/978-1-60566-766-9.ch016

ence landscape, and to be used for facing several real-world problems where classical systems show their limits. When ML systems work in real-world domains, they must typically deal with features such as complexity, need for efficiency and continuous adaptation to change. Optionally, humans may need to understand the inferred models in order to check and validate them. While the numeric and statistical setting, traditionally studied in the literature, can ensure efficiency, the other requirements call for more powerful representation formalisms. First-Order Logic (FOL for short) representation and processing techniques are more suitable than attribute-value and propositional ones for dealing with complexity and providing human understandability of the inferred models; moreover, they can deal with change and adaptation as well.

This work presents a suite of symbolic FOL learning algorithms and systems, that can serve as larger system components for satisfying these requirements in accomplishing real-world tasks. Their FOL representation language allows to effectively and efficiently deal with complex domains, yielding uniform human-understandable descriptions for observations and models. The FOL learning system INTHELEX tackles all of these requirements in a supervised setting. Being based on an incremental algorithm, it can update and refine the inferred models, instead of learning new ones from scratch, in order to account for new available evidence. Interestingly, its incremental abilities are not limited to examples processing, but allow to add to the theory and handle even completely new classes as soon as their instances come up. Conversely, when new observations become available for which a target classification is not given, an unsupervised setting is needed. In such a case, another module of the suite, that implements a recently developed similarity framework for FOL (Horn clause) representations, allows to face the problem.

This chapter focuses on Document Processing and Management, a real-world application domain that is gaining increasing interest in recent years, due to the progressive digitization of information sources. It involves almost all of the above requirements: need for quick response to the users' requests, complexity due to the high variability of documents, need for the librarians of checking and validating the inferred models, continuous flow of new documents. DOMINUS is a general-purpose document management framework that is able to process documents in standard electronic formats in order to recognize the type they belong to and their significant components based on their layout structure, and to selectively extract relevant information to be used for semantic indexing and later retrieval. Possible specific applications of the framework include support for the Semantic Web on Internet documents and content-based document management in organizations and libraries. Its architecture includes a module that provides Machine Learning services that support the different tasks involved in document processing and management. The FOL techniques presented in this chapter were embedded in such a module to provide the core functionality for making the framework powerful and flexible enough to deal with real-world cases.

In the following, after presenting the tasks and problems involved in Digital Libraries management, and how they have been tackled in DOMINUS, a more technical presentation of the incremental FOL framework exploited for document classification and understanding will be provided. The basics of the first-order logic setting will be recalled, and the similarity assessment on which the framework is based will be presented. The supervised and unsupervised learning modules and their cooperation will be then discussed, followed by experiments on a real-world dataset that show how the proposed framework can successfully and effectively be exploited as a viable solution to the incremental extension of documents and document classes in a digital library.

THE DOCUMENT PROCESSING DOMAIN

Recent evolution of computer technology in the last decades has provided the basis for exploiting computers as the principal source and repository of documents, and the Internet as the principal means of distribution, exchange and access for them. The consequent shift from paper to digital support for documents provided new opportunities for their management and processing activities, solving the problems of duplication and sharing that seriously affected legacy (paper) documents. However, making document production easy and cheap, it also introduced new problems, consisting in a huge amount of available documents (and in an associated decrease of their content quality) (Sankar, 2006). The well-known *information overload* problem, indeed, consists of the users' difficulty in accessing interesting information in large and heterogeneous repositories. Hence, the need to organize documents in a way that enables and makes easier, efficient and effective their retrieval and browsing based on an overview of the documents in the collection and of their relationships.

The primary criterion for searching interesting documents is by content. Hence, the documents in the repository should be grouped and organized accordingly. However, doing this manually is very expensive, and doing it automatically is very difficult due to the need of capturing document meaning (i.e., semantics). A more tractable starting point is exploiting layout similarity (i.e., syntax). The two approaches are often complementary, since documents of the same type/class usually have a similar spatial organization of their components and, conversely, different kinds of content are typically carried by documents having different layout styles. An additional, interesting and potentially useful relationship is that significant content is often placed in particular layout components, so that being able to identify the typical components of each group allows to selectively read only those components for identifying the document content. As a consequence, the ability to handle and manage documents according to layout structure similarity can be a key factor towards the ability to reach content-based management as well. Accordingly, document processing systems typically carry out two phases to identify the significant components of a digital document (Nagy, 2000). The former, called *Layout Analysis*, aims at identifying the geometrical organization of the components in a document page and at detecting relations among them, resulting in the so-called *Layout Structure*. The latter, called *Document Image Understanding*, aims at associating the proper logical role to each component, resulting in the so-called *Document Logical Structure*. Based on the assumption that the logical structure is different for different kinds of documents, Document Image Understanding can take advantage in terms of efficiency and accuracy if a particular sub-task, called *Document Image Classification*, is preliminarily carried out, aimed at identifying the document class (e.g., newspaper, scientific paper, email, technical report, call for papers) before associating each significant layout components for that class to a tag that expresses its role (e.g., signature, object, title, author, abstract, footnote, etc.).

Most work in the existing literature concerns the application of intelligent techniques to identify the document layout. In this chapter we focus on the identification of the proper layout class of a document, and of the main layout components of interest in it. Manual processing of the documents in the collection is clearly infeasible, not only, as already pointed out, for economic reasons, but also in principle, because of the huge amount of documents to be processed and, more seriously, due to the continuing extension of the document base which is typical in dynamic environments such as real-world Digital Libraries. This motivates the research for efficient and effective automatic techniques for document classification and understanding.

A first characteristic of the document domain is the wide variety of existing layout styles. This requires a powerful and flexible representation and processing framework, that allows to express and compare documents with different number of components, related to each other in many different ways. Attribute-Value representations can describe a document by means of a fixed set of features, each of which takes a value from a corresponding pre-specified value set. But one cannot know in advance how many components make up a generic document, and classes of documents are characterized by the presence of peculiar components and by the specific way their mutual positions and alignments are organized, rather than by their size and absolute position, that may vary even significantly. Some examples: the length of the title block could range from 1 to 4 lines, which would result in very different position of other components (such as authors and abstract) in the page, making their range of placement very wide consequently; in a scientific paper, it is useful to know that the acknowledgements usually appear above the references section and in the end of the document, or that the affiliation of the authors is reported generally at the beginning of the document, below or on the right of their names. Thus, document layout classes are hardly captured by static models, and classical attribute-value formalisms are not suitable for this task. Conversely, relational representations can capture the page layout complexity, allowing to identify invariants that characterize document classes, and additionally producing human-readable descriptions that can be exploited by domain experts for validation or understanding purposes. For these reasons, we propose to apply incremental FOL ML techniques along these phases of document processing where the classical statistical and numerical approaches to classification and learning may fail, being not able to deal with the lack of a strict layout regularity in the variety of documents available on-line.

Another issue to be considered when dealing with the document domain is the continuous flow of new and different documents in a Web repository or Digital Library. This means that any system working in this environment must be able to deal with changes in the domain, the context, or the user needs (in a word, it must be *incremental*). Traditional ML methods work according to the so-called batch approach, that requires the set of training examples, belonging to a defined number of classes, to be entirely available since the beginning, and exploits them altogether to produce a model. Thus, they assume that the classes are known and fixed since the beginning as well. If any change happens, they must restart the entire learning process to produce a model capable of coping with the new scenario. This prevents the opportunity of dealing with new instances, possibly belonging to new classes, that the learned theory does not account for. Conversely, the ability to revise a domain theory instead of restarting from scratch would be more appropriate to enable continuous responsiveness to the changes in the context, thus dealing with concept evolution, and could significantly improve efficiency. The incremental setting implicitly assumes that the information (observations) gained at any given moment is incomplete, and thus that any learned theory could be susceptible of changes. Hence incremental learning, as opposed to classical batch one, is needed whenever either incomplete information is available at the time of initial theory generation, or the nature (and the kinds) of the concepts evolves dynamically. In the document processing scenario, this would happen when the typing style of documents in a given class changes in time or when a brand new class is considered in the document repository.

RELATED WORK

Much of the work that has been done on document image analysis refers to algorithms and techniques that are applied to images of documents in order to obtain a computer-readable description from a scanned

document (Nagy, 2000). Even recently, the work on document analysis involves techniques that operate on image documents and a variety of statistical, rule-based, grammar-based techniques have been proposed as briefly reported in the following.

In (van Beusekom, 2006; van Beusekom, 2007) a new class of distance measures for document layouts is defined, based on a two-step procedure: after computing the distances between the blocks of two layouts, the blocks of one layout are assigned to the blocks of the other layout in a matching step. However, for layout structure discovering well-known layout algorithms from the literature are exploited, such as Voronoi (Kise 1998; Lu 2005), XY-Cut (Nagy, 1992), Run Length Smearing Algorithm (Wong, 1982), Docstrum (O’Gorman, 1993), Whitespace (Baird, 1994; Breuel, 2000). Sometimes, ideas in classical algorithms are enhanced with techniques for improving parameter setting, as in (Shi, 2005).

In (Marinai, 2006) a system for the retrieval of documents on the basis of layout similarity of document images is described. The system extracts the layout regions and represents them as an XY tree (Nagy, 1992). The indexing method combines a tree clustering algorithm based on the Self Organizing Maps (SOM - a particular type of artificial neural network that computes, during learning, an unsupervised clustering of the input data) with Principal Component Analysis. In such a way, similar pages are retrieved without comparing directly the query page to each indexed document. SOMs are applied in document image processing also for layout clustering aimed at document retrieval and page classification (Marinai, 2008). Other works exploit Artificial Neural Networks in document layout analysis and recognition as showed in (Marinai, 2005) or for logical document structure extraction, as in (Rangoni, 2006) extended in (Belaid, 2008). The document structure is described along a layer architecture. Each layer represents successive steps of the interpretation decomposition from physical (input) to logical (output) level of the document. The recognition process proceeds by repetitive perceptive cycles propagating information through the layers.

Other techniques exploit statistical approaches. In (Laven, 2005) a statistical pattern recognition approach to the problem of physical and logical layout analysis is investigated. The authors proposed an algorithm based on a logistic regression classifier working on a set of manually segmented and labelled page images, followed by statistical classifiers for the logical layout analysis. A similar approach can be found in (Carmagnac, 2004a) in which a semi-supervised document image classification system is presented. Given a set of unknown document images, the system uses unsupervised clustering to obtain grouping hypotheses for the same physical layout images. Then the operator can correct or validate them according to an objective, e.g. to have homogeneous groups of images whose descriptions are used for the training of the supervised document image classifier (Carmagnac, 2004) that is able to find the class of unknown documents. Another approach based on multiple hypotheses is proposed by (Kagehiro, 2008), where different hypotheses for layout segmentation, generated by low-level image analysis, are exploited by a final analysis aimed at character recognition.

A stream of works has concerned layout analysis of office documents, such as invoices, exploiting rule-based methods (Dengel, 2007), Case-Based Reasoning (Hamza, 2007) or formalized descriptions that can be compiled into executable code (Tuganbaev, 2005). Others have focused on collections of heterogeneous documents (Chen, 2007).

However, few works have faced the problem of discovering the layout and logical structure of documents in electronic formats, as opposed (but complementary) to document image analysis. (Seki, 2007) analyses document structure for simultaneous management of information in documents from various formats (image, PDF, and HTML). Most other contributions aim at extracting (some part of) the document content by means of a syntactic parsing of the PDF (Futrelle, 2003; Chao, 2003; Ramel, 2003) or

at discovering the background by means of statistical analysis applied to the numerical features of the documents and its components (Chao, 2004).

Other approaches based on domain knowledge are reported in (Hadjar, 2004; Rigamonti, 2005), where an expert provides a model for each class of documents to be handled. Such a model is represented by means of a grammar, i.e. a hierarchy of logical entities. Thus, after a step of syntactic parsing of the PDF document, aimed at discovering document primitives such as text entities, images and graphics, and after grouping the homogeneous entities discovered, the logical layout structure is recognized by labeling text entities (e.g., title, author, body) projecting them in the provided document model. A similar approach which uses grammars to annotate document components is proposed in (Anjewierden, 2001). Here, based on the model provided by an expert, a set of possible roles is assigned to each layout object. Then, they are collected into more complex objects until the logical structure is produced.

Processing of digital documents not having a strict layout is also faced by other works, such as Web pages (Feng, 2005; Burget, 2007; Guo, 2007) or e-mail messages (Chao, 2005).

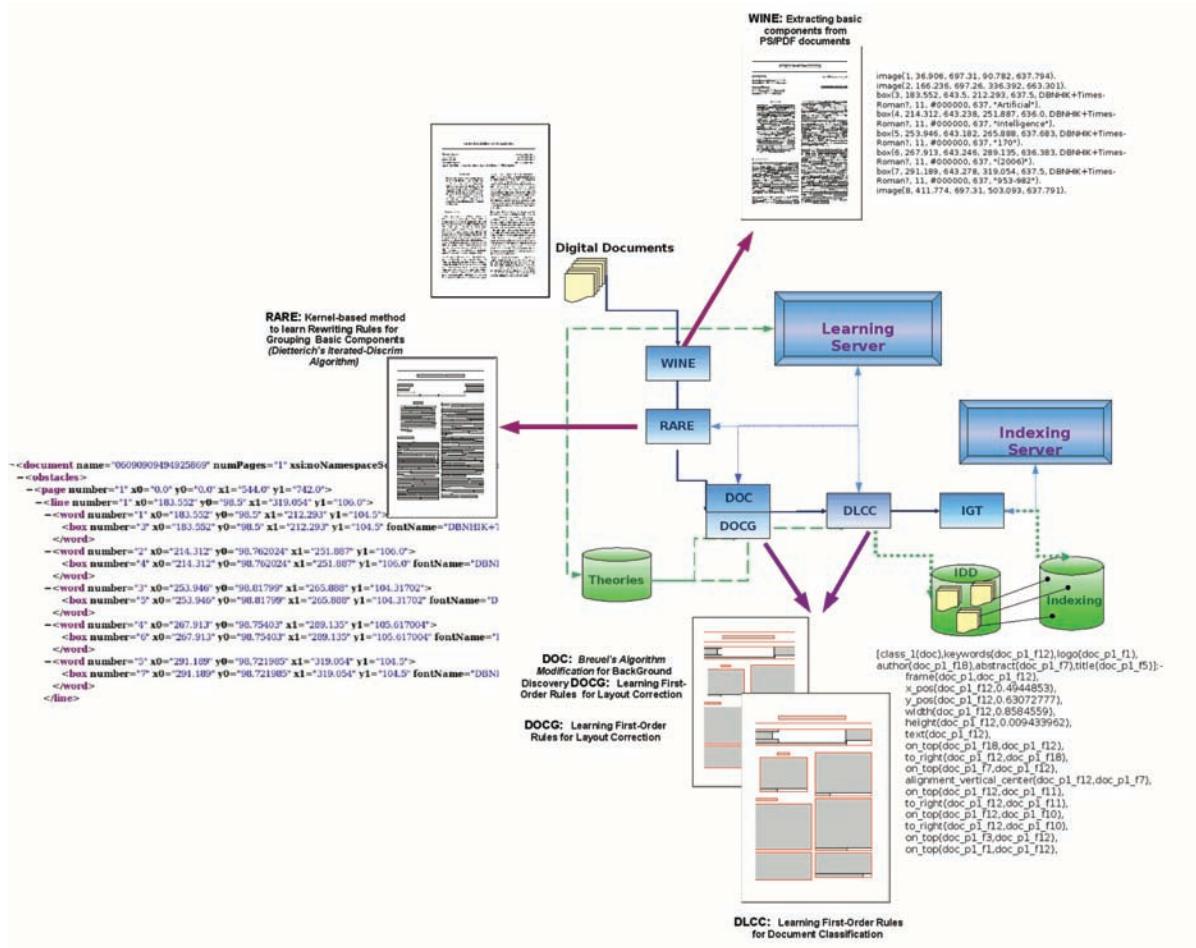
THE DOMINUS FRAMEWORK

DOMINUS (DOcument Management INtelligent Universal System) is a document processing framework characterized by the intensive exploitation of intelligent techniques in all involved tasks, from acquisition to analysis, indexing, categorization, storing and retrieval (Esposito, 2008). Its architecture, reported in Figure 1 along with the document processing flow, is general and flexible, so that it can be embedded as a document management engine into different Digital Library systems. A central role in DOMINUS is played by the Learning Server, which intervenes during different processing steps in order to continuously adapt the acquired knowledge taking into consideration new experimental evidence and changes in the context. The models inferred by the different components embedded in the Learning Server are stored for future exploitation in the Theories knowledge base.

The document layout analysis process starts with the application of a pre-processing module, called *WINE* (Wrapper for the Interpretation of Non-uniform Electronic document formats), that takes as input a digital document, translates it in an intermediate vector format that is independent on the original document format, and then produces the document's XML basic representation. Such a representation describes the document as a set of pages made up of basic blocks, often corresponding to graphical lines or rectangles, images and (groups of) text characters. Due to the large number of basic blocks, they are preliminarily aggregated in a bottom-up fashion: first, based on overlapping or adjacency criteria, composite blocks corresponding to whole words are produced; then, a further aggregation of words into lines is performed, based on inter-word spacing size. Using a fixed threshold (e.g., the mean distance between blocks) for this aggregation could be ineffective on multi-column documents, possibly merging into a single line words belonging to co-linear lines in adjacent columns. This problem was tackled through ML techniques, by casting the task to a Multiple Instance Problem and solving it by means of the kernel-based method proposed in (Dietterich, 1997). Such a method was embedded in the Learning Server, and exploited to generate rewriting rules for setting some parameters that are able to properly group word blocks into lines and will be exploited by *RARE* (Rule Aggregation REwriter).

After the line-blocks have been identified and represented by extending accordingly the XML description of the document, layout analysis proceeds top-down for collecting the semantically related blocks into consistent frames. The page background is identified first, and then the frames that surround docu-

Figure 1. DOMINUS Architecture and Processing Flow



ment content are derived as the complement of such a background. The *DOC* (Document Organization Composer) module, in charge of accomplishing this task, carries out the background structure analysis according to a modification of the algorithm proposed in (Breuel, 2002) to find iteratively the maximal white rectangles in a page. The original algorithm had to be modified in order to improve its efficiency and to stop when the significant frames are isolated, avoiding the retrieval of insignificant background white spaces such as inter-word or inter-line ones. Due to the great variability in layout both among and within documents, an all-purpose step where forcing termination of the process before its natural conclusion cannot be identified *a priori*. Rather, *DOC* applies some heuristics to decide, at each loop, whether exiting or continuing. Nevertheless, due to the background rectangles size and distribution, it can happen that the best stop point yields a final layout that is not ideal, because some significant white spaces have not yet been recognized by the background analysis or, conversely, because other insignificant ones have already been retrieved. These single wrong elements must be specifically corrected, usually by a manual intervention of an expert. To avoid whenever possible the user intervention, another module called *DOCG* (Document Organization Correction Generator) was implemented, in charge of learning and applying correction rules to be automatically applied on this task. These rules are automatically

learned from previous manual corrections collected by the system during the intervention of the experts on some training documents. Since the significance of a background rectangle in order to include or exclude it from the final layout is influenced by its relative size and position with respect to the other content and background rectangles around it, a FOL learning approach is needed to infer rules for this task, and a corresponding component has been added to the Learning Server to accomplish this task.

Once the layout structure recognition step successfully terminates, the *Document Image Understanding* task must be started, in charge of recognizing the document class and of associating a semantic role to each significant frame. These two sub-tasks are strongly interrelated, since the kind of components to be expected and identified in a document depends on its class (e.g., title, authors and their affiliations, abstract and references are typical of scientific papers, while in a commercial letter one might look for the sender, object, date and signature). Hence, the document class must be identified first, in order to know which kinds of components are to be expected and located, and then each component is associated to a label that properly expresses its role in the document. This task is performed by exploiting again FOL theories, since the document class is identified according to the distribution of frames in the layout, and the role played by each frame is determined by its content and spatial relationships with respect to surrounding frames. In DOMINUS, the classification and labelling theories are automatically learned, and the *DLCC* (Document and Layout Components Classifier) module takes care of managing and handling them. In a continuously evolving domain such as a digital document repository, not only new documents, but even completely new (known or unknown) classes can be expected to show up sooner or later, and hence the ability to refine and extend an existing theory according to new evidence is a fundamental feature. Hence, when a document/component of a class already known to the system is wrongly classified, the corresponding definition can be generalized or specialized accordingly, without re-starting learning from scratch. In case the document belongs to a class not yet considered by the theory, but an expert can identify it, the set of classes can be extended automatically. In case the class is unknown, the document is put in stand-by for future automatic discovery of new classes, this way dynamically extending the set of known classes. The incremental framework according to which these modifications and extensions of the theories take place in case of failure will be examined more thoroughly in the next sections. In the following, we will present the similarity-based learning framework underlying all steps of incremental learning and embedded in the Learning Server module. Such a framework is general, and in the case of DOMINUS works on FOL descriptions of the documents layout structure by identifying the description components that are more similar and hence more likely to correspond to each other.

At the end of this step both the original document and its XML representation, now including description of words, lines and frames, and enriched with class information and components annotation, have been stored in the Internal Document Database, IDD. Further steps, that are outside the scope of this work, concern text extraction from the significant components only and application of various Natural Language Processing, Information Retrieval and Information Extraction techniques that can support the semantic access to the document content. Specifically, indexing procedures are performed by the Indexing Server, useful for an effective content-based document retrieval. The *IGT* (IndexinG of Text) module currently includes full-text indexing, Latent Semantic Indexing, Keyword Extraction: the former supports classical retrieval, the second allows to retrieve documents that are related to a query although they do not contain exactly the same terms, and the latter aims at restricting the query focus only on key terms of the document.

FIRST-ORDER LOGIC PRELIMINARIES

FOL is a powerful formalism that can overcome the typical limitations shown by propositional or attribute-value representations by being able to express relations between objects. Logic Programming (Lloyd, 1987) is a FOL-based paradigm for making logic a machinable formalism. Logic programs are made up of *Horn clauses*, i.e. logical formulae of the form $l_1 \wedge \dots \wedge l_n \Rightarrow l_0$, often written in Prolog notation as $l_0 :- l_1, \dots, l_n$, to be interpreted as “ l_0 (*head*) is true, provided that l_1 and ... and l_n (*body*) are all true”. The l_i ’s are *literals*, where a literal is an *atom* (a predicate applied to terms) or its negation. A term is a constant, a variable or a function symbol applied to other terms. A predicate or function is a k -ary symbol (or has arity k) if it applies to k terms. Program execution is cast to finding a proof for some query expressed as a conjunction of literals. Differently from attribute-value representations, in FOL there is no fixed representation of an observation, and hence no one-to-one mapping between attributes when comparing two representations can be assumed. Thus, different portions of one description can be mapped onto different portions of another, a phenomenon known as *indeterminacy* that causes serious computational problems both when inducing models and when exploiting them for prediction. A *substitution* is a mapping from variables to terms. The classical generalization model adopted in Logic Programming is θ -subsumption, according to which a clause C is more general than another clause D ($C \leq_\theta D$) if and only if there exists a substitution θ such that, applied to terms of C , yields a subset of D ($C\theta \subseteq D$). The algorithm for finding the (unique) *least general generalization* (*lgg*) under θ -subsumption between two clauses was introduced by Plotkin (1970).

Datalog (Ceri, 1990) is a restriction of logic programming, born to deal with relational databases, in which only variables and constants are allowed as terms (thus, nesting by means of functions is not allowed). Generic clauses can be translated into Datalog ones and vice-versa (Rouveiro, 1992). The Object Identity assumption (OI for short) requires that different terms must denote different objects of the domain (it is generally assumed for constants, but the OI framework applies to variables as well). Datalog^{OI} is a restriction of Datalog in which clauses are interpreted under OI, which does not cause loss in expressive power (Semeraro, 1998). OI induces a new generalization model between clauses, θ_{OI} -subsumption, whose associated space is not a lattice (as under classical θ -subsumption), with the consequence that uniqueness of the *lgg* is not guaranteed but the clause structure is fixed since two different literals can never collapse because of their variables being bound on the same terms (which makes them more mechanizable).

Given a clause C , we define its *associated graph* as a Directed Acyclic Graph $G_C = (V, E)$, *stratified* (i.e., with the set of nodes partitioned) with

$$\begin{aligned} V &= \{l_0\} \cup \{l_i \mid i \in \{1, \dots, n\}, l_i \text{ built on } k\text{-ary predicate, } k > 1\} \\ E &\subseteq \{(a', a'') \in V \times V \mid \text{terms}(a') \cap \text{terms}(a'') \neq \emptyset\} \end{aligned}$$

(where $\text{terms}(f)$ denotes the set of terms that appear in the sub-formula f) built as follows. The head is the only node at level 0 (first element of the partition). Then, each successive element of the partition is made up by adding new nodes (not yet reached by edges) that have at least one term in common with nodes in the previous level. Hence, each node in the new level is linked by an incoming edge to each node in the previous level having among its arguments at least one term in common with it. The head literal, being unique, is exploited as a starting point in the graph (a kind of root) and provides a unique and well-defined perspective on the clause structure among the many possible, and hence significantly

reduces indeterminacy. For our purposes, there is no loss of generality in restricting to linked clauses (i.e., clauses whose associated graph is connected (Lloyd, 1987)): indeed, linked sub-parts of non-linked clauses can be dealt with separately, having no connection between each other.

SIMILARITY FRAMEWORK

Logic formalisms typically work on truth values, and thus only allow for binary (true/false) answers on whether a given (more general) formula accounts for another (more specific) one. Hence, the need for mechanisms to assess a degree of similarity among logic descriptions. Previous works on similarity/distance measures and techniques developed for comparing first-order descriptions are concerned with flexible matching (Esposito, 1992), supervised learning (Bisson, 1992a; Emde, 1996; Domingos, 1995; Sebag, 1997; Nienhuys-Cheng, 1998; Ramon, 2002; Kodratoff, 1986) and unsupervised learning (Thompson, 1989; Ramon, 1999; Bisson, 1992b; Blockeel, 1998). The similarity framework for FOL descriptions presented in the following overcomes some problems that are present in those works: it does not require assumptions and simplifying hypotheses (statistical independence, mutual exclusion) to ease the probability handling, no prior knowledge of the representation language is required and is not based on the presence of ‘mandatory’ relations, the user must not set weights on the predicates’ importance, it can be easily extended to handle negative information, it avoids the propagation of similarity between sub-components that poses the problem of indeterminacy in associations, it yields a unique value as a result of a comparison, which is more understandable and comfortable for handling, it is based directly on the structure, and not on derived features.

Similarity Function

The first step in setting up a similarity framework is designing a similarity function to evaluate the similarity between two items i' and i'' based on the presence of common and different features among them (Lin, 1998), that can be expressed by the following parameters:

n , the number of features owned by i' but not by i'' (*residual* of i' wrt i'');

l , the number of features owned both by i' and by i'' ;

m , the number of features owned by i'' but not by i' (*residual* of i'' wrt i').

Intuitively, a larger l should increase the similarity value, while larger values of n and m should decrease it. A classical formula in the literature based on these parameters is that by Tverski (1977), but its behaviour in extreme cases (full similarity or no overlapping between items features) can be problematic, for which reason we developed a novel similarity function based on the above parameters and on an additional parameter α ($0 \leq \alpha \leq 1$) that weights the importance of either item:

$$sf(\alpha, i', i'') = sf(\alpha, n, l, m) = (l+1) (\alpha/(l+n+2) + (1 - \alpha)/(l+m+2)) \in]0,1[$$

(in the following, we will just write $sf(i', i'')$ and $sf(n, l, m)$, implicitly assuming $\alpha = 0.5$). Since FOL formulae are complex expressions, evaluating the above parameters in a comparison is not straightforward. Hence we propose to evaluate those parameters for progressively large subcomponents of FOL

formulae: for basic components the similarity is computed directly from the above formula, whereas for complex components the similarity comes from a composition of the formula applied on their direct features with the similarity of their lower level sub-components involved. In FOL formulae, terms represent specific objects, that are related to each other by means of predicates. Accordingly, two levels of similarity can be defined for pairs of first-order descriptions: the *object* level, concerning similarities terms in the descriptions, and the *structure* one, referring to how the nets of relationships in the descriptions overlap.

Object Similarity

Let us consider two clauses C' and C'' . When comparing a pair of terms a' taken from C' and a'' taken from C'' , two kinds of object features can be distinguished: the properties they own (*characteristic features*), usually expressed by unary predicates (e.g. *female(X)*), and the roles they play (*relational features*), generally expressed by the position the object holds among n -ary predicate arguments (indeed, different positions refer to different roles played by the objects: e.g., in *mother(X,Y)* the first argument position identifies the role of the mother and the second one represents the role of the child).

If P' and P'' are the sets of characteristic features of a' and a'' , respectively, the *characteristic similarity* between a' and a'' is computed as $\text{sf}_c(a', a'') = \text{sf}(n_c, l_c, m_c)$ for the following parameters:

$$\begin{aligned} n_c &= |P' \setminus P''| \text{ number of properties owned by } a' \text{ in } C' \text{ but not by } a'' \text{ in } C''; \\ l_c &= |P' \cap P''| \text{ number of common properties between } a' \text{ in } C' \text{ and } a'' \text{ in } C''; \\ m_c &= |P'' \setminus P'| \text{ number of properties owned by } a'' \text{ in } C'' \text{ but not by } a' \text{ in } C'. \end{aligned}$$

Similarly, if R' and R'' are the multisets (indeed, one object can play many times the same role in different atoms: e.g., a mother of many children) of relational features of a' and a'' , respectively, the *relational similarity* between a' and a'' is computed as $\text{sf}_r(a', a'') = \text{sf}(n_r, l_r, m_r)$ for the following parameters:

$$\begin{aligned} n_r &= |R' \setminus R''| \text{ how many times } a' \text{ plays in } C' \text{ role(s) that } a'' \text{ does not play in } C''; \\ l_r &= |R' \cap R''| \text{ number of times that both } a' \text{ in } C' \text{ and } a'' \text{ in } C'' \text{ play the same role(s);} \\ m_r &= |R'' \setminus R'| \text{ how many times } a'' \text{ plays in } C'' \text{ role(s) that } a' \text{ does not play in } C'. \end{aligned}$$

Overall, the *object similarity* between two terms is defined as

$$\text{sf}_o(a', a'') = \text{sf}_c(a', a'') + \text{sf}_r(a', a'') \in]0, 2[.$$

Structural Similarity

When checking for the structural similarity of two formulae, many objects can be involved, and hence n -ary atoms expressing their mutual relationships represent a constraint on how each of them in the former formula can be mapped onto another in the latter. This is the most difficult part, since relations are specific to the first-order setting and are the cause of indeterminacy. Since we want to compare any two (sub-)formulae, in the following we will consider term *associations* as an extension of substitutions

that map terms onto terms, and call *compatible* two FOL (sub-)formulae that can be mapped onto each other without yielding inconsistent term associations (i.e., a term in one formula cannot be mapped onto different terms in the other formula).

Given an n -ary atom, we define its *star* as the multiset of n -ary predicates corresponding to the atoms linked to it by some common term (indeed, a predicate can appear in multiple instantiations among these atoms). Intuitively, the star of an atom depicts ‘in breadth’ how it relates to the rest of the formula. The *star similarity* between two compatible n -ary atoms l' and l'' having stars S' and S'' , respectively, is computed based on the number of common and different elements in each of the two stars, represented by the following parameters:

$$\begin{aligned} n_s &= |S' \setminus S''| \text{ how many more relations } l' \text{ has in } C' \text{ than } l'' \text{ has in } C''; \\ l_s &= |S' \cap S''| \text{ number of common relations between } l' \text{ in } C' \text{ and } l'' \text{ in } C''; \\ m_s &= |S'' \setminus S'| \text{ how many more relations } l'' \text{ has in } C'' \text{ than } l' \text{ has in } C'. \end{aligned}$$

Since atoms include terms as arguments, the object similarity of the involved terms must be considered as well, for all pairs of terms included in the association θ that map l' onto l'' of their arguments in corresponding positions:

$$sf_s(l', l'') = sf(n_s, l_s, m_s) + \text{Avg}(\{sf_o(t', t'')\}_{t'/t'' \in \theta}) \in]0,3[.$$

Moreover, we can consider all possible paths starting from the head and reaching *leaf* nodes (those with no outgoing edges) in the graph associated to the clauses: being such paths uniquely determined gives a leverage for significantly reducing the amount of indeterminacy in the comparison. Intuitively, a path in a clause depicts ‘in depth’ a given portion of the relations it describes. Paths can be interpreted as the basic components of the clause structure, and exploited instead of single atoms when checking similarity between clauses.

Given two paths $p' = \langle l'_0, l'_1, \dots, l'_{n'} \rangle$ in G_C and $p'' = \langle l''_0, l''_1, \dots, l''_{n''} \rangle$ in $G_{C''}$, their *intersection* is defined as the pair of longest compatible initial subsequences of p' and p'' , excluding the head:

$$p' \cap p'' = (\langle l'_1, \dots, l'_{k'} \rangle, \langle l''_1, \dots, l''_{k''} \rangle) \text{ s.t. } \forall i = 1, \dots, k: l'_0, l'_1, \dots, l'_{i-1} \text{ compatible with } l''_0, l''_1, \dots, l''_{i-1} \wedge (k = n' \vee k = n'' \vee l'_{i+1}, l'_{i+2}, \dots, l'_{k+1} \text{ incompatible with } l''_0, l''_1, \dots, l''_{k+1})$$

Hence, the *path similarity* between p' and p'' is computed according to the following parameters:

$$\begin{aligned} n_p &= n' - k \text{ length of the trail incompatible sequence of } p' \text{ wrt } p''; \\ l_p &= k \text{ length of the maximum compatible initial sequence of } p' \text{ and } p''; \\ m_p &= n'' - k \text{ length of the trail incompatible sequence of } p'' \text{ wrt } p' \end{aligned}$$

by considering also the star similarity values for all pairs of atoms associated by the initial compatible sequences:

$$sf_p(p', p'') = sf(n_p, l_p, m_p) + \text{Avg}(\{sf_s(l'_i, l''_i)\}_{i=1, \dots, k}) \in]0,4[.$$

Clause Similarity

Clause similarity is to be interpreted as follows: given two clauses, whose head literals contain the same number of arguments, the similarity between these two tuples of terms is computed as the similarity between the two bodies describing them. As for stars and paths, to assess the overall similarity between two clause bodies we need to evaluate the three parameters both globally and specifically for each body component, and then to merge the outcomes in a unique formula. As to the global parameters, they can be applied both to the number of common and different literals in the two clauses and to the number of common and different terms between them. To define what “common” means in this case, we exploit the concept of least general generalization (lgg), i.e. the most specific model for the given pair of descriptions. After computing the lgg, this is considered the common part in which counting the number of common literals and objects, while the literals and objects not covered by the lgg will be the residuals. Then, the similarity between each pair of literals of the two clauses that correspond to the same literal in the generalization can be taken into account as well.

More formally, given two clauses C' and C'' , call $C = \{l_1, \dots, l_k\}$ their lgg, and consider the substitutions θ' and θ'' such that $\forall i = 1, \dots, k: l_i \theta' = l'_i \in C'$ and $l_i \theta'' = l''_i \in C''$, respectively. Thus, the number of common and different atoms between the two clause is as follows:

$n = |C'| - |C|$ how many atoms in C' are not covered by its least general generalization with respect to C'' ;

$l = |C| = k$ maximal number of atoms that can be put in correspondence between C' and C'' according to their least general generalization;

$m = |C''| - |C|$ how many atoms in C'' are not covered by its least general generalization with respect to C' .

and the number of common and different objects is:

$n_o = |\text{terms}(C')| - |\text{terms}(C)|$ how many terms in C' are not associated by its least general generalization to terms in C'' ;

$l_o = |\text{terms}(C)|$ maximal number of terms that can be put in correspondence in C' and C'' as associated by their least general generalization;

$m_o = |\text{terms}(C'')| - |\text{terms}(C)|$ how many terms in C'' are not associated by its least general generalization to terms in C' .

Hence, the overall similarity between C' and C'' , can be computed according to a function called *formulae similitudo* and denoted fs , by considering also the star similarity values for all pairs of atoms associated by the least general generalization, as expressed by the following formula:

$$fs(C', C'') = sf(n, l, m) \cdot sf(n_o, l_o, m_o) + \text{Avg}(\{sf_s(l'_i, l''_i)\}_{i=1, \dots, k}) \in]0, 3[$$

that can obviously be normalized to $]0, 1[$ if needed. This function evaluates the similarity of two clauses according to the composite similarity of a maximal subset of their atoms that can be put in correspon-

dence (which includes both structural and object similarity), smoothed by adding the overall similarity in the number of atoms and objects between the two.

Similarity-Based Generalization

Implementing the theoretical *least general generalization* under θ_{or} -subsumption would be NP-hard, and hence too much computationally expensive. Thus, one might prefer to have a good approximation in acceptable time. To find such an approximation, the similarity between clause paths, as defined in previous sections, can be exploited, in order to quickly locate subcomponents of the clauses to be generalized that best match to each other. The algorithm, presented in (Ferilli, 2007) and reported below, works as follows: generalizations of all couples of paths in the original clauses to be generalized are scanned by decreasing similarity and added to the current partial generalization if compatible with it, or ignored otherwise. When all path generalizations have been considered, a generalization is obtained. Here, the former clause is taken as a pattern for the generalization, but in the end constants in such a pattern must be replaced with new variables (not yet appearing in the generalization, a different variable for each constant) to get the actual generalization. Although this is a typical greedy approach, backtracking can be performed to get more generalizations.

```

function lggor(E, C: clause): clause;
PC ← paths(C); PE ← paths(E);
P ← { (pC, pE) ∈ PC × PE | pC ∩ pE ≠ (< >, < >) } ;
G ← ∅; θ ← ∅
while (P ≠ ∅)
    (pC, pE) ← argmax(pC, pE) ∈ P (sf(pC, pE))
    P ← P \ { (pC, pE) }
    (gC, gE) ← pC ∩ pE
    θq ← mapping between terms s.t. gCθq = gE if (θq compatible with θ)
    G ← G ∪ gC; θ ← θ ∪ θq
    G ← replace all constants in G with new variables
return G

```

INCREMENTAL LEARNING FRAMEWORK

The proposed general incremental learning framework works as follows. Initially, a theory is provided, according to which classifying new observations that the system must process. Such a theory can be provided by an expert (the logical setting allows an easy formalization by humans thanks to its understandability), or have been previously learned by a ML system from examples, or even be empty (in which case the system must learn it instead of simply revising it). When the classifier is not able to recognize new incoming observations (which can be recognized because of no classification given, or multiple inconsistent classification, or a classification confidence under a given threshold, or a direct intervention of an expert), two cases can occur. If an expert can provide the correct classification, the system should modify accordingly the available theory. Such a modification can involve a refinement of the available definitions for known concepts, or require the addition of a brand new class in the theory

(which is a problem for all current ML systems, that require the whole set of concepts to be learned to be known since the beginning). Otherwise, if no class information can be provided for the problematic observation, it is put in stand-by until a sufficient number of rejections is reached, so that unsupervised learning can be run on them in order to automatically identify new classes according to their similarity. The *clustering* task aims at organizing a collection of unlabelled patterns into groups (clusters) of homogeneous elements based on their similarity. The similarity measure exploited to evaluate the distance between elements is responsible for the effectiveness of the clustering algorithms. Each discovered cluster becomes a new class, to be taken into account when considering new observations.

The supervised part of the above framework is carried out by INTHELEX (INcremental THEory Learner from Examples), a learning system for the induction of hierarchical theories from positive and negative examples. It is fully and inherently incremental: the initial theory can be empty and examples are considered and processed one by one according to a *closed loop* process, where feedback on performance (incorrectness of the theory on a new example) is used to activate the theory revision phase (Becker, 1985) in order to restore completeness and consistency. Theories learned by INTHELEX are Prolog programs implementing a classifier for the learned concepts. Examples in INTHELEX are definite ground Horn clauses, whose body describes the observation by means of only basic non-negated predicates of the representation language adopted for the problem at hand, and whose head reports the target class of a (tuple of) object(s) in the observation. In case of a negative example, the head is negated. A single observation may stand as an example or a counterexample for many concepts: a positive example for a concept is not considered as a negative example for all the other concepts (unless it is explicitly stated). A historical memory of all processed examples guarantees correctness of the future versions of the theory on the whole set of known examples.

INTHELEX is endowed with multiple inference strategies, according to the Inferential Theory of Learning framework (Michalski, 1994), in order to improve effectiveness and efficiency of the learning task. *Deduction* exploits the definitions in the learned theory and/or those in the Background Knowledge, if any, to recognize known objects in an example description. A dependency graph describes which concepts can contribute to the definition of which others. Whenever a new example is taken into account, its description is saturated with all instances of its sub-concepts in the dependency graph that can be recognized in its description. *Abstraction* allows to describe examples in a higher-level language to facilitate the generation of rules. In the abstraction framework adopted by INTHELEX (Zucker, 1998), abstraction operators defined in an Abstraction Theory, if any, can eliminate superfluous details, group specific component patterns into compound objects, reduce the number of object attributes, ignore the number of instances of a given object and obtain a coarser grain-size for attribute values. *Abduction* consists in hypothesizing unknown facts to be added to an observation, provided they are consistent with given integrity constraints, in such a way that the examples they represent are explained (if positive) or rejected (if negative) without modifying the current theory. INTHELEX adopts the abductive procedure by Kakas and Mancarella (Kakas, 1990), adapted to cope with OI.

As to *induction*, INTHELEX can learn simultaneously various concepts, possibly related to each other. When a new example is not correctly classified by the current theory, it is exploited by the *refinement operators* to fix the incorrect hypotheses. In particular, when a positive example is not covered, the theory can be revised in one of the following ways (listed by decreasing priority) such that completeness is restored:

- replacing a clause in the theory with one of its generalizations against the problematic example;
- adding a new clause to the theory, obtained by properly turning constants into variables in the problematic example (such a clause can even refer to a brand new concept);
- adding the problematic example as a positive exception.

When a negative example is covered, the theory consistency can be restored by performing one of the following actions (by decreasing priority):

- adding positive literals that characterize all the past positive examples of the concept (and exclude the problematic one) to the clauses that concur to the example coverage (starting from the lowest possible level);
- adding a negative literal that is able to discriminate the problematic example from all the past positive ones to the top-level clause in the theory by which the problematic example is covered;
- adding the problematic example as a negative exception.

Thus, examples are never rejected. Moreover, it does not need to know in advance what is the whole set of concepts to be learned: it learns a new concept as soon as examples about it are available. Thus, the algorithm implemented in INTHELEX allows to deal with two cases of incremental refinement of a theory: modification of the available hypotheses for known concepts (either extending or restricting their domains), and addition of hypotheses concerning new concepts not yet considered by the theory.

As to the unsupervised part of the framework, the proposed distance measure for FOL Horn Clauses can be exploited by any of the clustering techniques developed in the literature (Jain, 1999). As cluster prototypes, *medoid* must be exploited instead of centroids. The medoid of a cluster is defined as the observation in the cluster that has the minimum average distance from all the other members of the cluster, and is needed in a relational setting since first-order logic formulae do not induce an Euclidean space. Once the new clusters/class have been induced, three possibilities are available to assign new observations to one of them:

- such classes can be incorporated in the old theory, exploiting again INTHELEX to learn definitions for each of them according to the corresponding observations (*Conceptual Clustering*);
- otherwise, a k -Nearest Neighbour classification technique can be exploited, based on the same distance measure, in order to assign the new observation to the majority class among the closest instances;
- the new observation can be associated to the nearest medoid in the discovered classes.

EXPERIMENTS

The proposed strategy has been applied to real-world cases of document collections managed by DOMINUS. The system was run under Microsoft WindowsXP Professional™ on a PC endowed with a 2.13 GHz Intel processor and 2GB RAM. The dataset consisted in 353 scientific papers in PostScript or Portable Document Format, whose first pages layout descriptions were automatically generated by DOMINUS. According to these descriptions, the objective was identifying the papers' series and significant components. The documents belong to 4 different classes: Elsevier journals, Springer-Verlag

Lecture Notes (SVLN) series, Journal of Machine Learning Research (JMLR) and Machine Learning Journal (MLJ). Processing such a dataset involves a number of difficulties, due to several considerably complex aspects which were purposely introduced when choosing the documents to be included. First of all, the layout styles of the classes are quite similar to each other, which forces the automatic system to be able to grasp subtle differences in order to properly group them in distinct classes. Moreover, on the computational complexity side, efficiency is stressed since the 353 documents are described with a total of 67920 atoms, which yields an average of more than 192 atoms per description (however, some particularly complex layouts required more than 400 atoms). Lastly, the description language extensively exploits an inclusion relation between layout components that increases indeterminacy and hence can stress significantly the similarity computation. Runtime for the computation of the similarity between two observations in this dataset is on average of about 2sec, which is a reasonable time considering the descriptions complexity and the fact that the prototype has not been optimized in this preliminary version. A sample (short) document description is the following:

```
[not(icml(ferilli02)), svln(ferilli02), not(elsevier(ferilli02)),
not(ecai(ferilli02))]:-
    first_page(ferilli02, ferilli02_p1),
    frame(ferilli02_p1, ferilli02_p1_f5), text(ferilli02_p1_f5),
    to_right(ferilli02_p1_f1, ferilli02_p1_f5),
    on_top(ferilli02_p1_f5, ferilli02_p1_f1),
    on_top(ferilli02_p1_f12, ferilli02_p1_f5),
    to_right(ferilli02_p1_f12, ferilli02_p1_f5),
    on_top(ferilli02_p1_f3, ferilli02_p1_f5),
    to_right(ferilli02_p1_f3, ferilli02_p1_f5),
    on_top(ferilli02_p1_f2, ferilli02_p1_f5),
    to_right(ferilli02_p1_f2, ferilli02_p1_f5),
    on_top(ferilli02_p1_f4, ferilli02_p1_f5),
    to_right(ferilli02_p1_f4, ferilli02_p1_f5),
    frame(ferilli02_p1, ferilli02_p1_f12), text(ferilli02_p1_f12),
    to_right(ferilli02_p1_f1, ferilli02_p1_f12),
    on_top(ferilli02_p1_f12, ferilli02_p1_f1),
    center_vertical_aligned(ferilli02_p1_f12, ferilli02_p1_f1),
    on_top(ferilli02_p1_f3, ferilli02_p1_f12),
    center_vertical_aligned(ferilli02_p1_f12, ferilli02_p1_f3),
    to_right(ferilli02_p1_f2, ferilli02_p1_f12),
    on_top(ferilli02_p1_f12, ferilli02_p1_f2),
    center_vertical_aligned(ferilli02_p1_f12, ferilli02_p1_f2),
    to_right(ferilli02_p1_f4, ferilli02_p1_f12),
    on_top(ferilli02_p1_f12, ferilli02_p1_f4),
    center_vertical_aligned(ferilli02_p1_f12, ferilli02_p1_f4),
    frame(ferilli02_p1, ferilli02_p1_f4), text(ferilli02_p1_f4),
    on_top(ferilli02_p1_f4, ferilli02_p1_f1),
    center_vertical_aligned(ferilli02_p1_f4, ferilli02_p1_f1),
    on_top(ferilli02_p1_f3, ferilli02_p1_f4),
```

```

to_right(ferilli02_p1_f4, ferilli02_p1_f3),
center_vertical_aligned(ferilli02_p1_f4, ferilli02_p1_f3),
on_top(ferilli02_p1_f4, ferilli02_p1_f2),
to_right(ferilli02_p1_f4, ferilli02_p1_f2),
center_vertical_aligned(ferilli02_p1_f4, ferilli02_p1_f2),
frame(ferilli02_p1, ferilli02_p1_f1), text(ferilli02_p1_f1),
on_top(ferilli02_p1_f3, ferilli02_p1_f1),
to_right(ferilli02_p1_f1, ferilli02_p1_f3),
center_vertical_aligned(ferilli02_p1_f1, ferilli02_p1_f3),
on_top(ferilli02_p1_f2, ferilli02_p1_f1),
to_right(ferilli02_p1_f1, ferilli02_p1_f2),
center_vertical_aligned(ferilli02_p1_f1, ferilli02_p1_f2),
frame(ferilli02_p1, ferilli02_p1_f2), text(ferilli02_p1_f2),
on_top(ferilli02_p1_f3, ferilli02_p1_f2),
to_right(ferilli02_p1_f2, ferilli02_p1_f3),
center_vertical_aligned(ferilli02_p1_f2, ferilli02_p1_f3),
frame(ferilli02_p1, ferilli02_p1_f3), text(ferilli02_p1_f3),
width_medium(ferilli02_p1_f5), height_very_very_small(ferilli02_
p1_f5),
width_medium_large(ferilli02_p1_f12), height_very_
small(ferilli02_p1_f12),
width_medium_large(ferilli02_p1_f3), height_very_small(ferilli02_
p1_f3),
width_large(ferilli02_p1_f2), height_medium(ferilli02_p1_f2),
width_very_large(ferilli02_p1_f1), height_large(ferilli02_p1_f1),
width_very_large(ferilli02_p1_f4), height_medium_small(ferilli02_
p1_f4),
pos_left(ferilli02_p1_f5), pos_center(ferilli02_p1_f1),
pos_center(ferilli02_p1_f12), pos_center(ferilli02_p1_f2),
pos_center(ferilli02_p1_f3), pos_center(ferilli02_p1_f4),
pos_upper(ferilli02_p1_f12), pos_upper(ferilli02_p1_f3),
pos_upper(ferilli02_p1_f4), pos_middle(ferilli02_p1_f2),
pos_middle(ferilli02_p1_f5), pos_lower(ferilli02_p1_f1).

```

where clause body contains the actual description of the *ferilli02* document's first page layout, and the head states this document belongs to class *svln* (Springer Verlag Lecture Notes series), and does not belong to the other classes ICML, Elsevier, ECAI.

As to supervised learning, it was started with an empty theory. This means that no class was initially known and all classes in the final theory result from incremental revisions. Two versions of INTHELEX were run and compared to each other: the classical one (I), and a new one endowed with the similarity-guided generalization (SF) of (Ferilli, 2007). It is interesting to note that, on the document dataset, the similarity-driven generalization was very effective, preserving on average more than 90% atoms of the shortest clause, with a maximum of 99.48% and just 0.006 variance. This confirms the ability of the similarity technique to guide the generalization, so that the theoretical least general one is strictly ap-

Table 1. Document classification results.

	JMLR		Elsevier		MLJ		SVLN	
	SF	I	SF	I	SF	I	SF	I
Time	589	1962	53	304	3213	2975	399	663
Acc	0.98	0.98	1	0.99	0.96	0.93	0.98	0.94
F1	0.97	0.97	1	0.97	0.94	0.91	0.97	0.93

proximated and often exactly caught. The experiment on learning classification theories was set up as a 10-fold cross-validation, whose results according to different parameters are reported in Table 1. The similarity-driven version outperformed the classical one in all considered parameters:

- runtime
- learning behaviour (number of alternative definitions per concept, number of exceptions, number of generalizations/specializations performed)
- predictive accuracy (ratio of correctly classified examples): $(TP + TN) / (TP + FP + TN + FN)$
- F-measure, a weighted average of the following two indicators (in our experiments equal weight was assigned to both, a setting known as F1-measure):
 - Precision (ratio of correctly classified positive examples to the total number of examples classified as positive): $TP / (TP + FP)$
 - Recall (ratio of correctly classified positive examples to the number of actual positive examples): $TP / (TP + FN)$

where:

- TP (True Positive): number of positive examples predicted as positive by the theory;
- FP (True Positive): number of negative examples predicted as positive by the theory;
- TN (True Positive): number of negative examples predicted as negative by the theory;
- FN (True Positive): number of positive examples predicted as negative by the theory.

In the whole experiment, the similarity-driven version saved 1.15 hours, resulting in averages of 98% for predictive accuracy and 96% for F1-measure (+1% and +2%, respectively, with respect to the classical version). The high value of F-measure, in particular, is very encouraging, since it ensures that the technique behaves equally well on both positive and negative examples, although the latter are three times the former. A supervised experiment was run also for the understanding task. Specifically, the following labels were considered significant and searched for, where applicable: Title, Abstract, Author and Keywords, Logo. Overall averages are slightly worse than the classical version: 95% for accuracy and 89% for F-measure (-1% and -2%, respectively, with respect to the old version), but with huge runtime savings (between 1/3 and 1/2 of the time in single folds, for a total amount of 76.46 hours – i.e. 3.19 days! – overall). Since the proposed similarity framework is made up of a new similarity function and a similarity composition strategy, we have also checked whether the strategy alone, or the similarity function as well, are responsible for the good performance. For this purpose, we replaced the novel function with other well-known measures in the literature (Jaccard's, Tverski's and Dice's) in the proposed

Table 2. Document clustering results

	Elsevier	SVLN	JMLR	MLJ	TOTAL
Prec (%)	80	98.46	90.48	97.46	91.60
Rec (%)	100	85.33	100	87.79	93.28
Pur (%)					92.35

strategy. The new measure produced improvements up to +5.48% for precision, up to + 8.05% for recall and up to +2.83% for accuracy, which confirmed its actual contribution to advance the state-of-the-art. Here is a sample definition learned for classification:

```
ecai(A) :-  
    first_page(A, B), frame(B, C), text(C), height_very_very_  
    small(C),  
    on_top(D, C), text(D), pos_upper(D), frame(B, D).
```

to be read as “A document *A* belongs to class ECAI if in its first page *B* it contains two text frames, *C* and *D*, the latter in upper position in the page and above the former, having very very small height”.

As to the unsupervised setting, a classical *K*-means algorithm based on the same similarity computation was exploited on the same dataset, by hiding each document’s class to the system and asking it to identify 4 clusters (that would hopefully correspond to the actual document classes). The stop criterion was set as the moment in which the output of a new iteration is equal to a partition already seen in previous iterations (Ferilli, 2008). Specifically, the conceptual clustering option was chosen: i.e., after clustering, definitions for each cluster were learned exploiting INTHELEX on the same descriptions, tagged with the cluster label to which they were assigned. To evaluate performance, i.e. whether the clustering procedure was able to autonomously infer the correct classes, we checked the quality of the result by measuring the overlapping of each group with the correct classes (*supervised clustering*) according to precision, recall and purity (informally, this expresses the overlapping between classes and clusters and can be considered for clustering what predictive accuracy is in supervised learning). To do this, we had to decide which class each cluster had to be compared to. In fact, for each cluster the precision-recall values were neatly high for one class and considerably low for all the others, thus the choice of which was the best-matching class to be associated to each cluster became straightforward. Results, summarized in Table 2, show overall values well above 90%, near to those obtained by supervised learning, which indicates that the proposed method is highly effective in recognizing the original classes.

More in-depth analysis reveals that clusters associated to Elsevier and JMLR classes include all of the correct documents, reaching 100% recall; precision is also very high for JMLR, still high but neatly

Table 3. k-Nearest Neighbour classification results

	Elsevier	SVLN	JMLR	MLJ	TOTAL
Acc (%)	100	89.70	90.03	100	94.73

lower in the case of Elsevier. On the other hand, the quality of clusters corresponding to SVLN and MLJ are very similar among them, but opposite to the former cases, with a high precision balanced by a slightly lower recall. In any case, it should be taken into account that the layout styles of some of the classes in the dataset are very similar. Overall clustering runtime was in the order of hours, but since this task is not frequently carried out one could think, in a real Digital Library environment, to run it off-line and make the new classes available only after the procedure has been accomplished.

To complete the similarity-based framework for document image understanding, we also ran experiments on instance-based classification. Specifically, a k-NN approach was run to check its viability by comparing its performance to that of the theories learned by INTHELEX on the same dataset. k was set to 17, that is the square root of the number of learning instances (9/10 of the whole dataset in a 10-fold cross-validation setting, hence about 317) according to usual practice. Although we chose an odd (and prime) k , it should be noticed that, since the task was run on a multi-class dataset, the nearest neighbours are not bound to a binary classification and ties are possible. The average predictive accuracy results for the 10 folds on each class are reported in Table 3.

These results, very close to those obtained by learning conceptual definitions for each class, mean that few documents were associated to the wrong class, and hence are a further confirmation that the distance technique is very effective in identifying the proper similarity features within the given descriptions. Actually, very often in the correct cases not just a tiny majority, but almost all of the nearest neighbours to the description to be classified were from the same class. Errors were concentrated in the SVLN and JMLR classes, where, however, high accuracy rates were reached. MLJ seems quite distinct from the other classes, while Elsevier, although well-recognizable in itself, is somehow in between JMLR and SVLN, which are also close to each other. Interestingly, wrong classifications concerning Elsevier are unidirectional: JMLR and SVLN are sometimes confused with Elsevier, but the opposite never happened. Conversely, in the case of JMLR and SVLN it is bidirectional, suggesting a higher resemblance between the two. As to the comparison to the concept-learning algorithm, it is interesting to note that, while high performance on Elsevier and low performance on SVLN are confirmed from the conceptual learning case, for MLJ and JMLR the behaviour of the two approaches is opposite, suggesting somehow complementary advantages of each. This can be explained with the fact that printed journals impose a stricter fulfilment of layout style and standards, and hence their instances are more similar to each other. Thus, the concept learning algorithm is more suitable to learn definitions that generalize on peculiar features of such classes.

CONCLUSION

Digital libraries management is gaining increasing attention in a world in which the amount of available documents is growing so quickly that finding the needed information is harder and harder. Many tasks involved in document management can be profitably faced by Artificial Intelligence and Machine Learning techniques. This chapter specifically deals with the document image understanding, where first-order logic representation formalisms can provide the flexibility needed to manage documents with very different and variable layout, and incremental approaches are needed to deal with the continuous extension of the knowledge base by means of new documents and new document classes. A similarity-based framework for supervised and unsupervised learning is presented, as embedded in the prototypical document management system DOMINUS, that is able to refine existing class definitions according to

new evidence, but also to autonomously extend the set of known or unknown classes whenever needed, without restarting from scratch the learning process.

Experimental results on a real-world document dataset concerning scientific papers confirm that the proposed framework can efficiently and effectively face the problem, and that first-order logic representations and incremental approaches are a viable solution to document image understanding, with performance above 90% for all tasks: supervised learning, clustering and k -Nearest Neighbour classification. This is very encouraging, considering that precision and recall are typically contrasting parameters, and especially in the perspective of the representation-related difficulties. Future work will concern experimentation on different document types, optimization of the technique for reducing runtime and further extension of the framework with other techniques that can improve the overall behaviour, such as numerical and statistical techniques for sub-tasks that do not require the full power of first-order logic and can provide more efficiency.

REFERENCES

- Anjewierden, A. (2001). AIDAS: Incremental logical structure discovery in pdf documents. In *Proceedings of the 6th International Conference on Document Analysis and Recognition* (pp. 374-378). Washington, DC: IEEE Computer Society.
- Baird, H. S. (1994). Background structure in document images. In H. Bunke, P. S. P. Wang, & H. S. Baird (Eds.), *Document image analysis* (pp. 17-34). Singapore: World Scientific.
- Becker, J. M. (1985). *Inductive learning of decision rules with exceptions: Methodology and experimentation*. Unpublished bachelor's dissertation, University of Illinois at Urbana-Champaign.
- Belaïd, A., & Rangoni, Y. (2008). Structure extraction in printed documents using neural approaches. In S. Marinai & H. Fujisawa (Eds.), *Machine learning in document analysis and recognition* (pp. 1-24). Berlin, Germany: Springer.
- Bisson, G. (1992a). Learning in FOL with a similarity measure. In W. R. Swartout (Ed.), *Proceedings of the 10th National Conference on Artificial Intelligence – AAAI-92* (pp. 82-87).
- Bisson, G. (1992b). Conceptual clustering in a first order logic representation. In *Proceedings of the 10th European conference on Artificial intelligence*. New York: John Wiley & Sons.
- Blockeel, H., Raedt, L. D., & Ramon, J. (1998). Top-down induction of clustering trees. In J. Shavlik (Ed.), *Proceedings of the 15th International Conference on Machine Learning*. San Francisco: Morgan Kaufmann.
- Breuel, T. M. (2002). Two geometric algorithms for layout analysis. In *Proceedings of the 5th Workshop on Document Analysis Systems*.
- Burget, R. (2007). Layout based information extraction from HTML documents. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 624-628). Washington, DC: IEEE Computer Society.

- Carmagnac, F., Héroux, P., & Trupin, E. (2004a). Multi-view hac for semi-supervised document image classification. In S. Marinai & A. Dengel (Eds.), *Proceedings of the 6th International Workshop on Document Analysis Systems* (pp. 191-200). Berlin, Germany: Springer.
- Carmagnac, F., Héroux, P., & Trupin, E. (2004b). Distance based strategy for document image classification. In *Advances in pattern recognition* (pp. 894-902). Berlin, Germany: Springer.
- Ceri, S., Gottlob, G., & Tanca, L. (1990). *Logic programming and databases*. Berlin, Germany: Springer.
- Chao, H. (2003). Graphics extraction in PDF document. In T. Kanungo, E. H. B. Smith, J. Hu, & P. B. Kantor (Eds.), *SPIE - The International Society for Optical Engineering. Volume 5010*, (pp. 317-325).
- Chao, H., & Fan, J. (2004). Layout and content extraction for pdf documents. In S. Marinai & A. Dengel (Eds.), *Proceedings of the 6th International Workshop on Document Analysis Systems* (pp. 213-224). Berlin: Springer.
- Chao, H., & Lin, X. (2005). Capturing the layout of electronic documents for reuse in variable data printing. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 940-944). Washington, DC: IEEE Computer Society.
- Chen, S., Mao, S., & Thoma, G. (2007). Simultaneous layout style and logical entity recognition in a heterogeneous collection of documents. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 118-122). Washington, DC: IEEE Computer Society.
- Dengel, A. (2007). Learning of pattern-based rules for document classification. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 123-127). Washington, DC: IEEE Computer Society.
- Dietterich, T. G., Lathrop, R. H., & Lozano-Perez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2), 31–71. doi:10.1016/S0004-3702(96)00034-3
- Domingos, P. (1995). Rule induction and instance-based learning: A unified approach. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (pp. 1226-1232). San Francisco, CA: Morgan Kaufmann.
- Emde, W., & Wettschereck, D. (1996). Relational instance based learning. In L. Saitta (Ed.), *Proceedings of the 13th International Conference on Machine Learning* (pp. 122-130).
- Esposito, F., Ferilli, S., Basile, T. M. A., & Di Mauro, N. (2008). Machine learning for digital document processing: From layout analysis to metadata extraction. In S. Marinai & H. Fujisawa (Eds.), *Machine learning in document analysis and recognition* (pp. 79-112). Berlin, Germany: Springer.
- Esposito, F., Malerba, D., & Semeraro, G. (1992). Classification in noisy environments using a distance measure between structural symbolic descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(3), 390–402. doi:10.1109/34.120333
- Feng, J., Haffner, P., & Gilbert, M. (2005). A learning approach to discovering Web page semantic structures. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 1055-1059). Washington, DC: IEEE Computer Society.

- Ferilli, S., Basile, T. M. A., Di Mauro, N., Biba, M., & Esposito, F. (2007). Similarity-guided clause generalization. In R. Basili, & M. T. Pazienza (Eds.), *Proceedings of the AI*IA-2007: Artificial Intelligence and Human-Oriented Computing* (pp. 278-289). Berlin, Germany: Springer.
- Ferilli, S., Basile, T. M. A., Di Mauro, N., Biba, M., & Esposito, F. (2008). Generalization-based similarity for conceptual clustering. In Z. W. Ras, S. Tsumoto, & D. Zighed (Eds.), *Mining complex data* (pp. 13-26). Berlin, Germany: Springer.
- Futrelle, R. P., Shao, M., Cieslik, C., & Grimes, A. E. (2003). Extraction, layout analysis and classification of diagrams in PDF documents. In *Proceedings of the 7th International Conference on Document Analysis and Recognition* (pp. 1007-1014). Washington, DC: IEEE Computer Society.
- Guo, H.-F., Mahmud, J., Borodin, Y., Stent, A., & Ramakrishnan, I. V. (2007). A general approach for partitioning Web page content based on geometric and style information. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 929-933). Washington, DC: IEEE Computer Society.
- Hadjar, K., Rigamonti, M., Lalanne, D., & Ingold, R. (2004). Xed: A new tool for extracting hidden structures from electronic documents. In *Proceedings of the 1st International Workshop on Document Image Analysis for Libraries* (p. 212). Washington, DC: IEEE Computer Society.
- Hamza, H., Belaïd, Y., & Belaïd, A. (2007). A case-based reasoning approach for invoice structure extraction. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 327-331). Washington, DC: IEEE Computer Society.
- Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys*, 31(3), 264–323. doi:10.1145/331499.331504
- Kagehiro, T., & Fujisawa, H. (2008). Multiple hypotheses document analysis. In S. Marinai & H. Fujisawa (Eds.), *Machine learning in document analysis and recognition* (pp. 1-24). Berlin, Germany: Springer.
- Kakas, A. C., & Mancarella, P. (1990). On the relation of truth maintenance and abduction. In *Proceedings of the 1st Pacific Rim International Conference on Artificial Intelligence*, Nagoya, Japan.
- Kise, K., Sato, A., & Iwata, M. (1998). Segmentation of page images using the Area Voronoi Diagram. *Computer Vision and Image Understanding*, 70(3), 370–382. doi:10.1006/cviu.1998.0684
- Kodratoff, Y., & Ganascia, J.-G. (1986). Improving the generalization step in learning. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach: Volume II* (pp. 215-244). Los Altos, CA: Kaufmann.
- Laven, K., Leishman, S., & Roweis, S. T. (2005). A statistical learning approach to document image analysis. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 357-361). Washington, DC: IEEE Computer Society.
- Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning* (pp. 296-304). San Francisco, CA: Morgan Kaufmann.
- Lloyd, J. W. (1987). *Foundations of logic programming* (2nd ed.). New York: Springer-Verlag.

- Lu, Y., & Tan, C. L. (2005). Constructing Area Voronoi Diagram in document images. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 342-346). Washington, DC: IEEE Computer Society.
- Marinai, S. (2008). Self-organizing maps for clustering in document image analysis. In S. Marinai & H. Fujisawa (Eds.), *Machine learning in document analysis and recognition* (pp. 381-408). Berlin, Germany: Springer.
- Marinai, S., Gori, M., & Soda, G. (2005). Artificial neural networks for document analysis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1), 23–35. doi:10.1109/TPAMI.2005.4
- Marinai, S., Marino, E., & Soda, G. (2006). Tree clustering for layout based document image retrieval. In *Proceedings of the 2nd International Workshop on Document Image Analysis for Libraries* (pp. 243-253). Washington, DC: IEEE Computer Society.
- Michalski, R. S. (1994). Inferential theory of learning. Developing foundations for multistrategy learning. In R. S. Michalski & G. Tecuci (Eds.), *Machine learning. a multistrategy approach, volume IV* (pp. 3-61). San Mateo, CA: Morgan Kaufmann.
- Nagy, G. (2000). Twenty years of document image analysis in PAMI. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), 38–62. doi:10.1109/34.824820
- Nagy, G., Seth, S., & Viswanathan, M. (1992). A prototype document image analysis system for technical journals. *Computer*, 25(7), 10–22. doi:10.1109/2.144436
- Nienhuys-Cheng, S. (1998). Distances and limits on Herbrand interpretations. In D. Page (Ed.), *Proceedings of the ILP-98*. Berlin, Germany: Springer.
- O’Gorman, L. (1993). The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11), 1162–1173. doi:10.1109/34.244677
- Plotkin, G. D. (1970). A note on inductive generalization. *Machine Intelligence*, 5, 153–163.
- Ramel, J. Y., Crucianu, M., Vincent, N., & Faure, C. (2003). Detection, extraction and representation of tables. In *Proceedings of the 7th International Conference on Document Analysis and Recognition* (pp. 374-378). Washington, DC: IEEE Computer Society.
- Ramon, J. (2002). *Clustering and instance based learning in first order logic*. Unpublished doctoral dissertation, K.U. Leuven, Belgium.
- Ramon, J., & Dehaspe, L. (1999). Upgrading Bayesian clustering to first order logic. In *Proceedings of the 9th Belgian-Dutch Conference on Machine Learning*, Department of Computer Science, K. U. Leuven.
- Rangoni, Y., & Belaid, A. (2006). Document logical structure analysis based on perceptive cycles. In H. Bunke, & A. L. Spitz (Eds.), *Proceedings of the 7th International Workshop on Document Analysis Systems* (pp. 117-128). Berlin, Germany: Springer.

- Rigamonti, M., Bloechle, J. L., Hadjar, K., Lalanne, D., & Ingold, R. (2005). Towards a canonical and structured representation of PDF documents through reverse engineering. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 1050-1055). Washington, DC: IEEE Computer Society.
- Rouveiro, C. (1992). Extensions of inversion of resolution applied to theory completion. In *Inductive logic programming* (pp. 64-90). Amsterdam: Academic Press.
- Sankar, K. P., Ambati, V., Pratha, L., & Jawahar, C. V. (2006). Digitizing a million books: Challenges for document analysis. In H. Bunke & A. L. Spitz (Eds.), *Proceedings of the 7th International Workshop on Document Analysis Systems* (pp. 425-436). Berlin, Germany: Springer.
- Sebag, M. (1997). Distance induction in first order logic. In N. Lavrač & S. Džeroski (Eds.), *In Proceedings of the ILP-97*. Berlin, Germany: Springer.
- Seki, M., Fujio, M., Nagasaki, T., Shinjo, H., & Marukawa, K. (2007). Information management system using structure analysis of paper/electronic documents and its applications. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 689-693). Washington, DC: IEEE Computer Society.
- Semeraro, G., Esposito, F., Malerba, D., Fanizzi, N., & Ferilli, S. (1998). A logic framework for the incremental inductive synthesis of datalog theories. In N. E. Fuchs (Ed.), *Logic program synthesis and transformation* (LNCS 1463, pp. 300-321). Berlin, Germany: Springer.
- Shi, Z., & Govindaraju, V. (2005). Multi-scale techniques for document page segmentation. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 1020-1024). Washington, DC: IEEE Computer Society.
- Thompson, K., & Langley, P. (1989). Incremental concept formation with composite objects. In *Proceedings of the 6th International Workshop on Machine Learning*. San Francisco: Morgan Kaufmann.
- Tuganbaev, D., Pakhchanian, A., & Deryagin, D. (2005). Universal data capture technology from semi-structured forms. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 458-462). Washington, DC: IEEE Computer Society.
- Tversky, A. (1977). Features of similarity. *Psychological Review*, 84(4), 327–352. doi:10.1037/0033-295X.84.4.327
- van Beusekom, J., Keysers, D., Shafait, F., & Breuel, T. M. (2006). Distance measures for layout-based document image retrieval. In *Proceedings of the 2nd International Workshop on Document Image Analysis for Libraries* (pp. 232-242). Washington, DC: IEEE Computer Society.
- van Beusekom, J., Keysers, D., Shafait, F., & Breuel, T. M. (2007). Example-based logical labeling of document title page images. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 919-923). Washington, DC: IEEE Computer Society.
- Wong, K. Y., Casey, R. G., & Wahl, F. M. (1982). Document analysis system. *IBM Journal of Research and Development*, 26(6), 647–656.

Zucker, J.-D. (1998). Semantic abstraction for concept representation and learning. In R. S. Michalski & L. Saitta (Eds.), *Proceedings of the 4th International Workshop on Multistrategy Learning* (pp. 157-164).

KEY TERMS AND THEIR DEFINITIONS

Artificial Intelligence: “the science and engineering of making intelligent machines” (J. McCarthy).

Inductive Logic Programming: a subfield of Machine Learning which uses Logic Programming as a uniform representation for examples, background knowledge and hypotheses.

Incremental Learning: a learning algorithm is incremental if it can process training examples that become available over time, usually one at a time, modifying the learned theory accordingly if necessary without restarting from scratch.

Clustering: for the aims of this work, we deliberately focus on one definition of clustering, that is the partitioning of a data set into subsets (clusters), so that the data in each subset (ideally) share some common trait – often proximity according to some defined distance measure.

Document Processing: in the scope of this work, the conversion of typed and handwritten text and images on paper-based & electronic documents into electronic information that can be easily searched and inter-related.

Document Image Understanding: “the formal representation of the abstract relationships indicated by the two-dimensional arrangement of the symbols” (G. Nagy).

Digital Libraries: a library in which collections are stored in digital formats and accessible by computers

Chapter 17

Machine Learning and Financial Investing

Jie Du
UMBC, USA

Roy Rada
UMBC, USA

ABSTRACT

This chapter presents the case for knowledge-based machine learning in financial investing. Machine learning here, while it will exploit knowledge, will also rely heavily on the evolutionary computation paradigm of learning, namely reproduction with change and selection of the fit. The chapter will begin with a model for financial investing and then review what has been reported in the literature as regards knowledge-based and machine-learning-based methods for financial investing. Finally, a design of a financial investing system is described which incorporates the key features identified through the literature review. The emerging trend of incorporating knowledge-based methods into evolutionary methods for financial investing suggests opportunities for future researchers.

INTRODUCTION

The field of machine learning applied to financial investing is complicated but promises opportunities as well as challenges. The speculative financial investing problem, compared with other financial problems, such as financial accounting, attracts substantial research interest. This chapter introduces the case for machine-learning-based methods to be applied to the financial investing problem. The objective of the chapter is to be a state-of-the-art review upon which a model for research on intelligent financial investing system can be built.

A conceptual model for financial investing is presented, and relevant literature is reviewed as regards knowledge-based and machine-learning-based methods for financial investing. The literature review reveals three patterns. Knowledge-based methods, such as expert systems, were mostly used to solve financial

DOI: 10.4018/978-1-60566-766-9.ch017

problems in the early 1990s, while machine-learning-based methods, such as evolutionary computation, gained the dominate position in this field in the early 2000s. The most frequently addressed financial topic is ‘Asset Valuation’. The trend is to incorporate knowledge-based technologies into evolutionary computation for financial investing.

Later in this chapter the design of an intelligent financial investing system is presented which incorporates the key features identified by the literature review. More specifically, this chapter is organized as follows. The ‘Background Section’ introduces the problem of financial investing and presents a model for financial investing systems. The procedure used in the literature review is given in the ‘Method Section’. The detailed literature review and the key features identified through the literature review are presented in the ‘Results’ and ‘Analysis’ Sections. The penultimate section contains the design of the intelligent financial investing system. The concluding remarks and future trends are contained in the ‘Conclusion Section’.

BACKGROUND

Development in the State of the Art of Machine Learning

Machine learning can be defined as a program that based on **experience E** with respect to some class of **tasks T** and a **performance measure P** improves its performance at **task T**, as measured by **P**, with **experience E** (Mitchell, 1997). Machine learning systems are not directly programmed to solve a problem, instead they develop based on examples of how they should behave and from trial-and-error experience trying to solve the problem.

The field of machine learning addresses the question of “how can we build computer systems which can automatically improve with experience, and what are the fundamental laws that govern all learning processes?” (Mitchell, 2006).

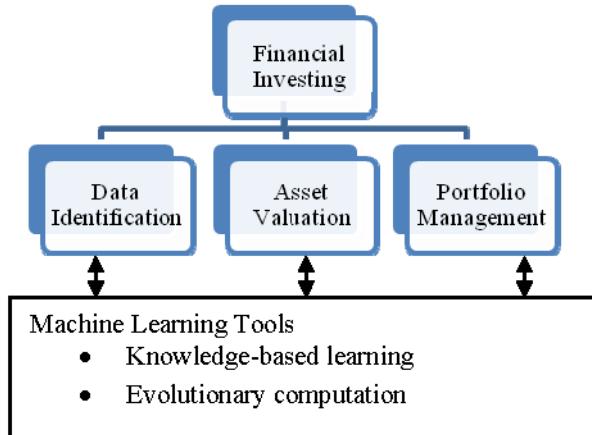
Different techniques are used in different subfields of machine learning, such as neural networks(Chauvin & Rumelhart, 1995), instance-based learning (Aha, Kibler & Albert, 1991), decision tree learning (Quinlan, 1993), computational learning theory (Kearns & Vazirani, 1994), genetic algorithms(Mitchell, 1996), statistical learning methods (Bishop, 1996), and reinforcement learning (Kaelbling, Littman & Moore, 1996). In recent years, machine learning has shed light on many other disciplines, such as economics, finance, and management. Many machine learning tools, including neural networks and genetic algorithms, are used in intelligent financial investing research.

The Problem of Financial Investing

Financial investing systems can be seen as having three phases: 1) data collection, 2) asset valuation and 3) portfolio management. Financial investing systems should follow these steps(Kingdon, 1997):

- Identify and collect useful data which can be used to make predictions,
- Forecast assets’ future value, and
- Optimize a portfolio.

Figure 1. Conceptual model of financial investing system



The phase of data collection identifies what data might be used to predict an asset's value. The asset value is influenced by many factors. For stock valuation, quantitative factors, such as the individual company's financial attributes and historical stock price, are often used in predicting the future stock price. Wu et al (2006) use the history of stock prices and national money supply. Wang and Chan (2006) analyze the history of stock prices with a two-layer bias decision tree.

On the other hand, qualitative factors, such as the corporation's management quality and market position may play an important role in future corporate value. Kim and Lee(1995) use the quality of a company's management and its financial policies as input data to perform the task of bond rating for those companies. They built an expert system to collect some of this subjective information. Expert systems technologies are widely used in asset valuation. Akoka et al (1994) present an expert system used for assessing the chances of success of a new product.

Portfolio management starts with a set of assets and generates a weighted combination of assets to maximize return for any particular risk. Abdelazim and Wahba (2006) use genetic algorithms and neural networks for portfolio management.

Financial Investing and Machine Learning

A conceptual model for the application of machine learning to financial investing addresses financial data input selection, asset valuation, and portfolio management (see Figure 1).

What changes have occurred over time in the artificial intelligence tools used for financial applications? Are knowledge-based methods and evolution-based methods being integrated? Three hypotheses about the literature follow:

- **Hypothesis 1:** Knowledge-based methods were mostly used to solve financial problems in the early 1990s, while machine-learning-based methods gained the dominate position in this field in the early 2000s.
- **Hypothesis 2:** 'Asset Valuation' is the most frequently addressed financial concept, and

- **Hypothesis 3:** Incorporating knowledge-based technologies into evolutionary computation for financial investing is an emerging trend.

METHOD

Relevant literature was reviewed to test the hypotheses about research trends. More specifically, two sources were systematically analyzed(Rada, 2008; Rada & Ha, 2008):

- One is the journal *Expert System with Applications*, and
- The other is the IEEE literature.

The *Expert Systems with Applications* journal was chosen because of its strength in financial applications, including accounting, economics, finance, and stock trading. The IEEE literature was chosen because of its focus on technology. Admittedly, many other journals may offer articles relevant to the topic of machine learning and investing, but the chosen sample of journals should provide useful evidence of trends. Detailed information on the procedure, query selection, and data analysis are presented next.

Procedure

This literature review had two stages. The first stage focused on individual sources, the second stage focused on a combination of sources. For both the journal *Expert System with Applications* and the IEEE literature database, the query requested documents that contained either the string ‘financ’ or ‘invest’.

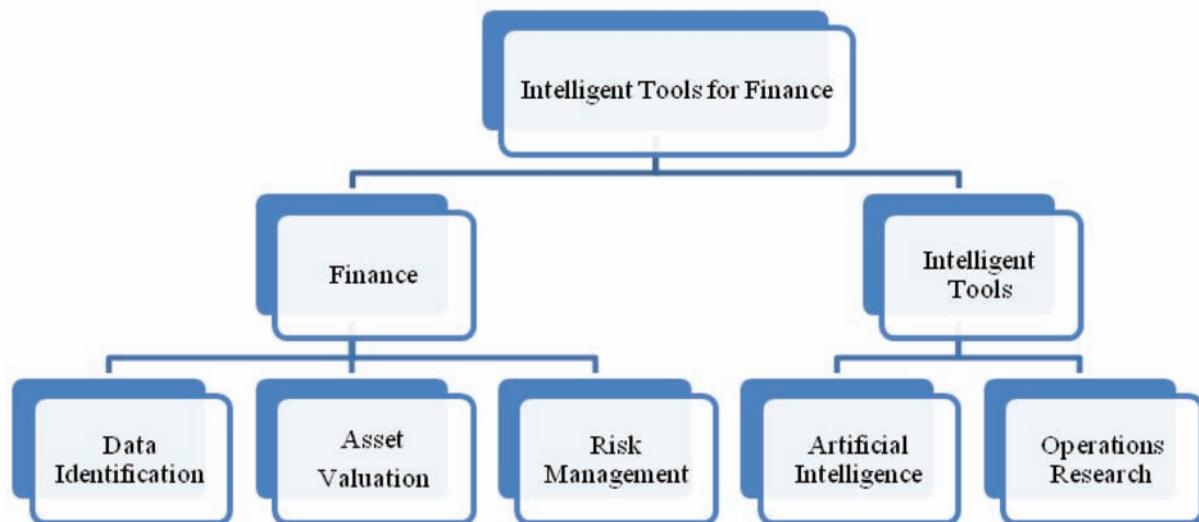
To detect chronological trends, certain time periods were used. More specifically, for *Expert System with Applications*, the time periods were 1991-1996 or post-2005. For the IEEE literature, the time periods were 1988-1993 or 2000-2006.

Each of the references was studied to see whether or not it was relevant to the application of intelligent tools to finance. If the abstract lacked information, the full text of the article was retrieved and analyzed. References which were not relevant were discarded. As a result, 65 articles from *Expert System with Applications* were used for future study. 71 articles from the IEEE literature were used for future study. In summary, 136 articles relevant to application of intelligent tools to finance were analyzed for this literature review.

Indexing

The classification language used for this study was borrowed from(Rada, 2008). The classification language has a root node of the hierarchy of ‘Intelligent Tools for Finance’. The root node has two children: ‘Finance’ and ‘Intelligent Tools’. ‘Finance’, in turn, has three children: 1) data identification, 2) asset valuation, and 3) risk management. ‘Intelligent Tools’ has two children: 1) artificial intelligence, and 2) operations research. The top three levels of the classification language are shown in Figure 2. For each reference, the most prominent ‘Finance’ concept and the most prominent ‘Intelligent Tools’ concept were chosen respectively to index this reference.

Figure 2. The top three levels of the classification language



RESULTS AND ANALYSIS

Expert System with Applications

Among the 65 articles from *Expert System with Applications* literature, 29 belonged to the period from 1991 to 1996 and 36 belonged to the period from 2006 to 2007. First, ‘Intelligent Tools’ indexing was analyzed. Of the 29 articles from 1991 to 1996, 17 articles were indexed with ‘expert systems’ and two articles (Coakley, 1995; Jo & Han, 1996) were indexed with ‘evolutionary computing’. Of the 36 articles from 2006 to 2007, 13 articles were indexed with ‘evolutionary computing’ while only one article was indexed with ‘expert systems’. The pattern of changes from expert systems approaches on finance to evolutionary computing approaches on finance was found, and thus Hypothesis 1 was supported.

In the next step, ‘Finance’ indexing was analyzed. Of the 65 articles, 33 articles were indexed with ‘Accounting’ which was in the sub-tree of ‘Asset Valuation’. Four articles were indexed with ‘Stock Valuation’ which was also in the sub-tree of ‘Asset Valuation’. Thus, more than half of the 65 articles were focused on ‘Asset Valuation’ and thus Hypothesis 2 was supported.

Three of the 65 articles addressed the incorporation of financial knowledge into evolution. Tsakonas et al (2006) present the efficient use of hybrid intelligent systems consisting of logical rules and genetic programming for forecasting bankruptcy. Oh et al (2006) combine evolutionary computation technologies and knowledge-based technologies for portfolio optimization. Dempster and Leemans (2006) incorporate a knowledge-based portfolio management layer in a neural network. All of these articles were published after 2005. Thus Hypothesis 3, that is a trend of incorporating knowledge into evolutionary computing, was supported.

IEEE Literature

Among the 71 articles from the IEEE literature, 12 belonged to the period from 1988 to 1993 and 59 belonged to the period from 2000 to 2006. First, ‘Intelligent Tools’ indexing was analyzed. Of the 12 articles from 1988 to 1993, five articles were indexed with ‘expert systems’ while only one article was indexed with ‘evolutionary computing’. Of the 59 articles from 2000 to 2006, more than 10 articles were indexed with ‘evolutionary computing’, while only one article was indexed with ‘expert system’. Based on the results, the pattern of changes from expert systems approaches on finance to evolutionary computing approaches on finance was found, and thus Hypothesis 1 was supported.

In the next step, ‘Finance’ indexing was analyzed. Of the 71 articles, 48 articles were indexed with ‘Asset Valuation’ concept. Nine articles were indexed with ‘Risk Management’ concept. Three articles were indexed with ‘Data Identification’ concept. Thus, more than half of the 71 articles were focused on ‘Asset Valuation’ and thus Hypothesis 2 was supported.

In order to test the third hypothesis, articles which addressed the incorporation of financial knowledge into evolution were identified. Lajbcygier (2004) incorporates an option pricing model into the representation of a neural network. Bhattacharyya et al (2002) present the incorporation of semantic restrictions for Genetic Programming-based searching of trading decision models. Hung et al (2003) combine the adaptive supervised learning decision trading system with portfolio optimization knowledge. Three other papers that addressed the combination of knowledge-based methods and evolutionary computation methods were found. All of the six articles were published after 2000. Thus Hypothesis 3, that is a trend of incorporating knowledge into evolutionary computing, was supported.

Combination of the Two Sources

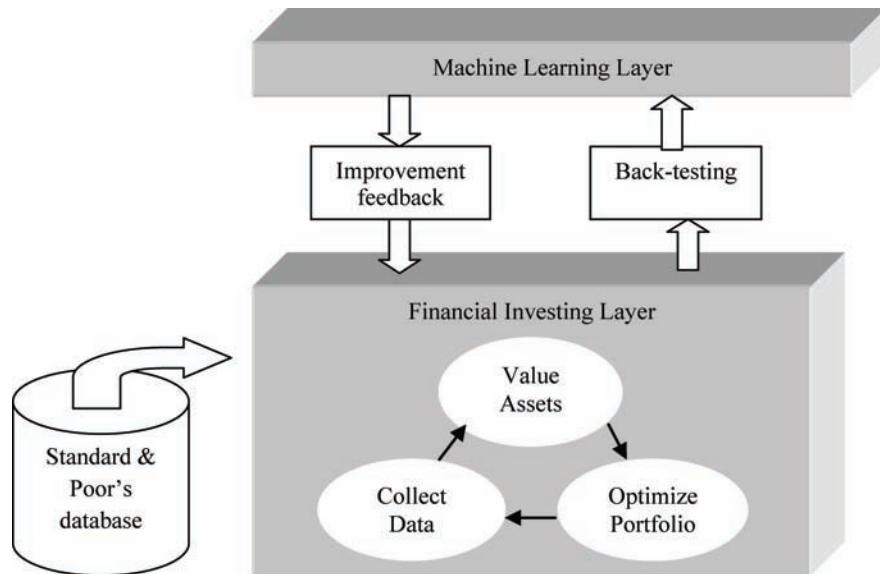
The three hypotheses were supported by both sources. More specifically, expert systems tools are mostly used in the 1990s, while evolutionary computing tools are most used in the 2000s. ‘Asset Valuation’ is the most frequently addressed topic compared with other financial areas. As expected, the literature shows a chronological trend of combining knowledge-based methods with evolutionary computing methods.

As financial investing systems need to identify data, value assets, and manage risk and these three phases have feedback loops, it is surprising that very little literature addresses the three phases. The most popular application phase for artificial intelligence in the literature is the asset valuation phase. A few papers have addressed both data identification and asset valuation and a few others have incorporated asset valuation and risk management. For instance, Kim (2006) studied data identification and asset valuation. Dempster and Leemans (2006) addressed asset valuation and risk management.

DESIGN OF THE FINANCIAL INVESTING SYSTEM

The design of a stock investing system is presented which can be used to support machine learning experiments. This system is called IIS for Intelligent Investing System. Some recent machine learning experiments from the literature are described to show how they could be built onto a stock investing platform such as IIS.

Figure 3. The structure of financial investing system



System Structure

An intelligent financial investing system design will illustrate the concepts abstracted from the literature. Identified by the key feature of incorporation of knowledge-based methods with evolutionary methods, the system has two layers: a financial investing layer and a machine learning layer. The structure of the financial investing system can be illustrated by Figure 3.

- The financial investing layer will 1) collect data about stocks from a Standard & Poor's database management system, particularly financial statements over multiple years, 2) value assets based on forecasting of financial statements, and 3) do portfolio optimization.
- The machine learning layer will make improvements in the investing layer. The design will show how ontologies about finance and about machine learning are exploited in making changes to the financial investing layer.

Data Collection

Many sources of financial data are available. Governments often make publically available large amounts of information about domestic productivity, financial reserves, and so on. Publicly-traded companies publish annual reports which contain prominently their financial statements. The challenge in part is to gain access to this financial data in a standard format that is readily machine-manipulable. One prominent source of a large amount of publically traded financial statement data is the company Standard & Poor's, and one of its popular datasets is called Compustat.

In IIS, the input data are retrieved from Standard & Poor's Compustat North America data set. That data set has financial statements from over 20,000 companies for multiple decades. The system allows a user to specify whether annual or quarterly data is wanted, a base year, the number of historical years to

use in trending, and how many time periods in the future to forecast. For example in some experiments, the base year is chosen as 2005, annual financial statements from the years 2001-2005 are retrieved, and forecasts are made for the year 2006. In a comparable experiment with quarterly data, the base is the 4th quarter of 2005, quarterly statements from 2001 to 2005 are collected and the financial statement for the first quarter 2006 is forecast.

Asset Valuation

In IIS the historical data are used for forecasting. Many forecasting techniques exist. One could use neural networks of various kinds to try to capture relationships among the data and thus predict future values based on past data. In finance the most popular approach to forecasting depends on least squares regression.

Least squares regression depends on assuming a functional form that fits the data and then determining the parameters of that function which give the smallest error when modeling the historical data. Namely, the square of the distance between an actual data point and the modeled data point is used. In one IIS test bed, four kinds of trending functions were used to capture hidden patterns in the historical data.

In addition to trending, other information is used to assist financial forecasting. Financial ratios can reflect a company's performance on profitability, solvency, and management. In IIS, a modified neural logic network based on (Matsatsinis, Doumpas & Zopounidis, 1997) was adapted to evaluate each company's financial performance. Fourteen financial ratios were computed based on the financial statement data and fed to the neural logic network. The output of this neural logic network is a 3-point company value of {1, -1, 0} corresponding to 'satisfactory', 'unsatisfactory' and 'do not know', respectively.

In IIS, this neural logic network is utilized to bias prediction. The learning algorithm modifies the weights on edges of the neural logic network. The best weights are defined as ones that obtain the highest correlation of the output of the neural logic network and the target fitness. For example, in one experiment, quarterly statements from 2005 for 11 petrochemical companies are collected and the earnings for the 1st quarter of 2006 are forecast. The target fitness is the difference between the actual earnings in 2006 and the IIS-predicted earnings in 2006.

Interactions between Asset Valuation and Learning

The interaction between the asset valuation layer and the machine learning layer has two aspects. For training, back-testing is conducted to see how well the forecasting of a financial attribute value in year $(b+1)$ matches the actual value in year $(b+1)$. The prediction error is defined as:

$$\text{Prediction error} = (\text{Forecasting value} - \text{Actual value})/\text{Actual value}$$

The trend function which has the smallest prediction error is deemed to be the best prediction solution and will be used for future prediction. For the testing dataset, the best prediction solution obtained from back-testing in the training dataset will be used to predict the future value.

Another approach to machine learning on the asset valuation part is to manipulate the financial ratio analysis portion. Tsakonas et al (2006) use evolutionary learning to modify the system architecture and the parameters used in the system.

The approach incorporates a neural logic network (Teh, 1995) with genetic programming (Koza, 1992). A formal grammar is adapted to generate neural logic networks. The input to the network is financial ratios. The output is a classification of a company as likely to go bankrupt or not. The structure of the network and the weights on edges are modified through a genetic programming process. The architecture is altered by entering nodes sequentially or in parallel onto an initial neural logic network to obtain the best possible topology of a neural logic network. The network weights are maintained to preserve the meaning of the neural net to people. Thus, understandable expert-rules and knowledge discovery are achieved.

Portfolio Management

Collecting data and analyzing the value of particular assets are only the first steps of intelligent investing. Deciding how much of what to buy and sell at any given moment is crucial. These buy and sell decisions are part of portfolio management. IIS has been used to experiment with portfolio management using evolutionary computation. Strings that included information about how much money the investor would put into each company were generated. These strings were fed to a genetic algorithm that modified the strings. Each new string was evaluated for the financial return it would produce over time. The system did show improvement over time. However, the field of portfolio management is well studied and has many techniques which merit consideration in an intelligent investing system that is supposed to combine knowledge and evolutionary computation.

IIS incorporates several portfolio optimization techniques and methods for testing their performance. Two of the most famous portfolio optimization techniques are those championed by Sharpe and by Merton. Covariances in stock earnings are computed and portfolios are recommended to achieve any particular mix of risk and reward. Both the Sharpe and the Merton methods of portfolio optimization have been implemented in IIS.

The portfolio recommendations from IIS can be studied with back-testing. For instance, stock data from 2001 to 2005 is collected, and forecasts of 2006 stock earnings are made. Portfolio optimization suggests what stocks to buy in 2006 based on the 2006 forecasts and past price histories of the stocks. The actual stock earnings that the portfolio would generate at the end of 2006 are determined and used to assess the performance of the program.

CONCLUSION

Patterns were found in the literature on artificial intelligence applied to financial investing. Knowledge-based methods, such as expert systems, were popular in the early 1990s, while machine-learning-based methods, such as evolutionary computation, were popular later. A trend is the incorporation of knowledge-based technologies into evolutionary computation for financial investing.

Based on a conceptual model of intelligent financial investing systems and features identified by the literature, the design of an intelligent financial investing system was proposed. Challenges for machine learning applied to financial investing include the following. First, as financial investing systems need to identify data, value assets, and manage risk and these three phases have feedback loops, it is surprising that very little literature addresses the three phases together. Future work should go from focusing on one aspect of the problem to a holistic approach to the problem. Second, combining evolution-based

approaches with knowledge-based approaches is a promising research trend. Third, developing knowledge bases that can be incorporated into evolutionary techniques is critical.

REFERENCES

- Abdelazim, H., & Wahba, K. (2006). An artificial intelligence approach to portfolio selection and management. *International Journal of Financial Services Management*, 1(2-3), 243–254.
- Aha, D., Kibler, D., & Albert, M. (1991). Instance-based learning algorithms. *Machine Learning*, 6(1), 37–66.
- Akoka, J., Leune, B., & Koster, A. (1994). Expert system for feasibility assessment of product development. *Expert Systems with Applications*, 7(2), 291–303. doi:10.1016/0957-4174(94)90045-0
- Bhattacharyya, S., Pictet, O. V., & Zumbach, G. (2002). Knowledge-intensive genetic discovery in foreign exchange markets. *IEEE Transactions on Evolutionary Computation*, 6(2), 169–181. doi:10.1109/4235.996016
- Bishop, C. M. (1996). *Neural networks for pattern recognition*. Oxford, UK: Oxford University Press.
- Chauvin, Y., & Rumelhart, D. (1995). *Backpropagation: Theory, architectures, and applications*. Hillsdale, NJ: Lawrence Erlbaum.
- Coakley, J. R. (1995). Using pattern analysis methods to supplement attention-directing analytical procedures. *Expert Systems with Applications*, 9(4), 513–528. doi:10.1016/0957-4174(95)00021-6
- Dempster, M. A. H., & Leemans, V. (2006). An automated FX trading system using adaptive reinforcement learning. *Expert Systems with Applications*, 30(3), 543–552. doi:10.1016/j.eswa.2005.10.012
- Hung, K.-k., Cheung, Y.-m., & Xu, L. (2003). An extended ASLD trading system to enhance portfolio management. *IEEE Transactions on Neural Networks*, 14(2), 413–425. doi:10.1109/TNN.2003.809423
- Jo, H., & Han, I. (1996). Integration of case-based forecasting, neural network, and discriminant analysis for bankruptcy prediction. *Expert Systems with Applications*, 11(4), 415–422. doi:10.1016/S0957-4174(96)00056-5
- Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4, 237–285.
- Kearns, M. J., & Vazirani, U. V. (1994). *An introduction to computational learning theory*. Cambridge, MA: MIT Press.
- Kim, B.-O., & Lee, S. M. (1995). Bond rating expert system for industrial companies. *Expert Systems with Applications*, 9(1), 63–70. doi:10.1016/0957-4174(94)00049-2
- Kim, K.-J. (2006). Artificial neural networks with evolutionary instance selection for financial forecasting. *Expert Systems with Applications*, 30(3), 519–526. doi:10.1016/j.eswa.2005.10.007
- Kingdon, J. (1997). *Intelligent systems and financial forecasting*. London: Springer-Verlag.

- Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural selection*. Cambridge, MA: MIT Press.
- Lajbcygier, P. (2004). Improving option pricing with the product constrained hybrid neural network. *IEEE Transactions on Neural Networks*, 15(2), 465–476. doi:10.1109/TNN.2004.824265
- Matsatsinis, N. F., Doumpos, M., & Zopounidis, C. (1997). Knowledge acquisition and representation for expert systems in the field of financial analysis. *Expert Systems with Applications*, 12(2), 247–262. doi:10.1016/S0957-4174(96)00098-X
- Mitchell, T. M. (1996). *An introduction to genetic algorithms*. Cambridge, MA: MIT Press.
- Mitchell, T. M. (1997). *Machine learning*. New York: McGraw Hill.
- Mitchell, T. M. (2006). *The discipline of machine learning*. Pittsburgh, PA: Carnegie Mellon University.
- Oh, K., Kim, T. Y., Min, S.-H., & Lee, H. Y. (2006). Portfolio algorithm based on portfolio beta using genetic algorithm. *Expert Systems with Applications*, 30(3), 527–534. doi:10.1016/j.eswa.2005.10.010
- Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco: Morgan Kaufmann.
- Rada, R. (2008). Expert systems and evolutionary computing for financial investing: A review. *Expert Systems with Applications*, 34(4), 2232–2240. doi:10.1016/j.eswa.2007.05.012
- Rada, R., & Ha, L. (2008). Intelligent technologies for investing: A review of engineering literature. *Intelligent Decision Technologies*, 2(3), 167–178.
- Teh, H.-H. (1995). *Neural logic networks*. Singapore: World Scientific.
- Tsakonas, A., Dounias, G., Doumpos, M., & Zopounidis, C. (2006). Bankruptcy prediction with neural logic networks by means of grammar-guided genetic programming. *Expert Systems with Applications*, 30(3), 449–461. doi:10.1016/j.eswa.2005.10.009
- Wang, J.-L., & Chan, S.-H. (2006). Stock market trading rule discovery using two-layer bias decision tree. *Expert Systems with Applications*, 30(4), 605–611. doi:10.1016/j.eswa.2005.07.006
- Wu, M.-C., Lin, S.-Y., & Lin, C.-H. (2006). An effective application of decision tree to stock trading. *Expert Systems with Applications*, 31(2), 270–274. doi:10.1016/j.eswa.2005.09.026

KEY TERMS AND DEFINITIONS

Assets: An asset is defined as a probable future economic benefit obtained or controlled by a particular entity as a result of a past transaction or event in business and accounting.

Portfolio: In finance, a portfolio is an appropriate mix of or collection of investments held by an institution or a private individual

Portfolio Management: Portfolio management involves deciding what assets to include in the portfolio by comparing the expected return from portfolios of different asset bundles, given the goals of the portfolio owner and changing economic conditions

Machine Learning: As a broad subfield of artificial intelligence, machine learning is concerned with the design and development of algorithms and techniques that allow computers to “learn”

Evolutionary Computation: In computer science evolutionary computation is a subfield of artificial intelligence (more particularly computational intelligence) that uses iterative progress in a population selected in a guided random search using parallel processing to achieve the desired end.

Neural Logic Network: A neural logic network is a finite directed graph, consisting of a set of input nodes and an output node

Genetic Algorithms: Genetic algorithms are search procedures based on the mechanics of natural selection and genetics and are in the class of evolutionary computation techniques.

Genetic Programming: Genetic programming is a machine learning technique used to optimize a population of computer programs according to a fitness landscape determined by a program’s ability to perform a given computational task.

Chapter 18

Applications of Evolutionary Neural Networks for Sales Forecasting of Fashionable Products

Yong Yu

The Hong Kong Polytechnic University, Hong Kong

Tsan-Ming Choi

The Hong Kong Polytechnic University, Hong Kong

Kin-Fan Au

The Hong Kong Polytechnic University, Hong Kong

Zhan-Li Sun

The Hong Kong Polytechnic University, Hong Kong

ABSTRACT

The evolutionary neural network (ENN), which is the hybrid combination of evolutionary computation and neural network, is a suitable candidate for topology design, and is widely adopted. An ENN approach with a direct binary representation to every single neural network connection is proposed in this chapter for sales forecasting of fashionable products. In this chapter, the authors will first explore the details on how an evolutionary computation approach can be applied in searching for a desirable network structure for establishing the appropriate sales forecasting system. The optimized ENN structure for sales forecasting is then developed. With the use of real sales data, the authors compare the performances of the proposed ENN forecasting scheme with several traditional methods which include artificial neural network (ANN) and SARIMA. The authors obtain the conditions in which their proposed ENN outperforms other methods. Insights regarding the applications of ENN for forecasting sales of fashionable products are generated. Finally, future research directions are outlined.

DOI: 10.4018/978-1-60566-766-9.ch018

INTRODUCTION

It is well-known that fashionable products exhibit a highly volatile demand pattern which follows the ever-changing market trend. In order to cope with the trend and market response, retailers would take measures such as accurate and quick response (Hammond 1990), automatic inventory replenishment, and even collaborative planning and forecasting between supply chain agents (De Toni 2000) are widely-adopted. It is known that fashion companies can improve their inventory and pricing decisions by acquiring market information (see Choi, 2007). By utilizing market information, fashion companies can reduce the forecasting error which in turn helps to lower stocking costs (e.g., see Thomassey, 2005). However, even with information updating, an efficient forecasting is still fundamentally important. In this chapter we will explore the use of a well-established machine intelligence tool, the Evolutionary Neural Networks, for forecasting the sales of fashionable products.

Essentially, the idea of Artificial Neural Network (ANN) was inspired by the biological neural networks and the mathematical modeling of how the human brain works. The early multilayered neural network with a training algorithm was introduced in 1970s (Fukushima, 1975). ANN has been applied successfully in a lot of areas such as pattern recognition of handwriting, speech, and sounds, etc. (Pao, 1989). Research has also proven that the multilayer feedforward neural networks are universal approximators (Hornik et al., 1989). This feature makes ANN a powerful tool for pattern classification and recognition. Owing to the similar requirement of pattern learning or approximation of historical data in forecasting, in the 1980s and 1990s, many researchers have proposed to use ANN for forecasting (White, 1988; Hill et al.; 1994, Kuan et al., 1995; Dorffner, 1996; Luxhoj et al., 1996; Frank et al., 2002; Schikora et al., 2003; Sztandera et al., 2004; Cavalieri et al., 2004; Wu et al., 1994 & 2004; Sun et al., 2008). As the ANN is a data-driven self-adaptive method, few a priori assumptions about the model are needed for problems under study. It can learn from past data and capture subtle functional relationships among the data even if the underlying relationships are unknown or hard to describe. After learning the data, ANN can often correctly infer other part in the data. This feature enables ANN to predict future behavior from examples of past behavior, and thus makes it an ideal tool for forecasting. ANN has more general and flexible functional forms than what the traditional statistical methods can effectively deal with. Traditional statistical forecasting models have limitations in estimating underlying functions due to the complexity of the real system, while ANN can be a good alternative method to identify these functions. ANN is also nonlinear (Chu, 2003), and it is capable of performing nonlinear modeling without a priori knowledge about the relationships between input and output variables. Thus it is a more general and flexible modeling tool for forecasting. In this book chapter paper, we discuss how an Evolutionary Neural Network (ENN) model can be utilized in assisting fashion companies to make a scientifically sound and implementable forecasting decision for the sales of fashionable products. A flexible overfitting test is explicitly proposed which has been shown to be capable of improving the performance of the existing ENN models. The performance of our proposed forecasting ENN method is compared with the traditional statistical model SARIMA, and the traditional ANN. It is found that our proposed ENN model exhibits significant benefits and outperforms SARIMA and ANN in terms of forecasting accuracy. Future research direction is outlined.

APPLICATIONS OF ANN IN FORECASTING

In sales forecasting with ANN, the neural network is trained with past data and then this network is used to predict future sales values. In the early stage of using ANN in forecasting, several ANN architectures have been proposed, such as radial-basis functions or recurrent networks, while the multilayer feedforward network (Lapedes et al., 1987; Cortez et al. 1995; Shtub et al., 1999; McMullen, 2001; Huang et al., 2004; Sun et al., 2007) prevailed at last. At the beginning, ANN could not surpass traditional statistical forecasting methods in many cases (Chatfield, 1993). Researchers began to realize that the topology of ANN was one of the key factors which influenced their learning capacities to various data. For instance, (Tang et al, 1995) claimed that for time series of different complexities there are optimal neural network topologies and parameters that enable them to learn more efficiently. However, there are problems on how to determine the best topology and parameter sets for an efficient learning process. Thus, to select and adopt a suitable neural forecasting model is always a challenging task of utilizing ANN in forecasting. ANNs are made of basic units arranged in layers. These basic units are sometimes called “neurons” (see Figure 1) (Abdi, 2003). A unit collects information provided by other units or external inputs to which it is connected with weighted connections called synapses. These weights, called synaptic weights, multiply (i.e., amplify or attenuate) the input information: A positive weight is considered excitatory, a negative weight inhibitory. Each of these units is a simplified model of a neuron and transforms its input information into an output response. Figure 2 illustrates a basic multilayer feedforward ANN.

The problem of overfitting is a common challenge for almost all kinds of forecasting models, so finding a parsimonious model for a specific problem has always been the strategy in forecasting systems. Therefore in practice, the fully connected network such as the one as shown in Figure 2 is often not an ideal model for forecasting. For ANN, finding a parsimonious model involves the determination of two crucial parameters, namely, the number of neurons and the connections between them. For example, (Poli et al., 1994) built a stochastic MLP model with random connections, and (Baum, et al., 1989 and Amirikian et al., 1994) discussed general relationship between the generalizability of a network and the size of the networks. Though the neural network topology design can be addressed by simple trial-and-error procedures such as exploring networks with a different number of hidden nodes, there are more elaborated methods such as the pruning (Thimm et al., 1995) and constructive (Kwok et al., 1999) algorithms which use hill-climbing procedures, but it is difficult for them to overcome the local minima problem. The evolutionary computation (EC) offers an alternative to these methods. The EC or the genetic algorithms are the optimization procedures which can mimic natural selection and biological evolution. By this approach the EC can help achieve a more efficient ANN topology for forecasting (Miller et al.,

Figure 1. The basic neural unit (neuron)

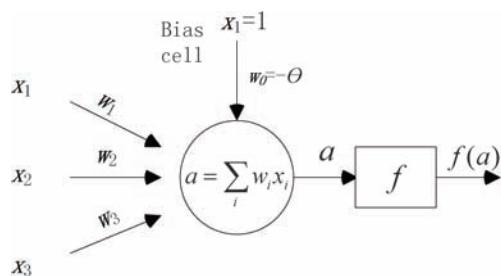
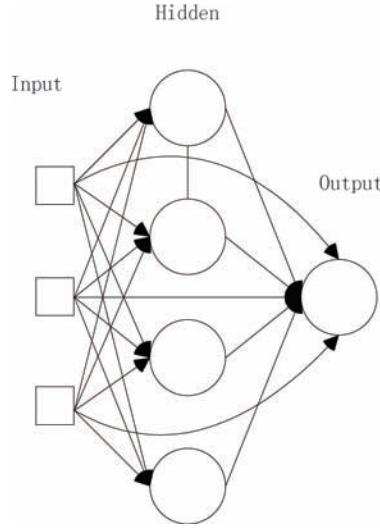


Figure 2. A fully connected three-layer feedforward ANN



1989; Guo and Uhrig, 1992; Jones, 1993; Schiffmann et al., 1993; Happel and Murre, 1994; Lee et al., 1997, Sun et al., 2006). Under EC, a global multi-point search can be conducted, which can quickly locate areas of high quality even when the search space is large and complex. The ENN (Yao, 1999), which is the hybrid combination of evolutionary computation and neural networks, is a suitable candidate for topology design, and is widely adopted (Falco et al., 1998; Chen et al., 1999). Unlike most of the ENN approaches which use indirect parametric representation, an ENN approach with a direct binary representation to every single neural network connection has been proposed for time series forecasting (Cortez et al., 2006). With the automatic pruning of the hidden nodes and input nodes with no connection, the numbers of input and hidden nodes are automatically decided. The Bayesian Information Criterion (*BIC*) is used as the fitness function to speed-up the searching process of ENN as in Cortez et al. (2006), and an overfitting test is also proposed to facilitate searching (Au et al. 2008).

In order to perform time series modeling with a multilayer perception, a set of training cases is created from the data by adopting a sliding time window. This combination is called “time lagged feed-forward network”. A sliding window is defined by the sequence $\langle k_1, k_2, \dots, k_i \rangle$, for a network with i inputs and time lags. The EC searching approach we adopted is a kind of pruning mechanism. Connections between neurons will be pruned in order to make the ANN structure suitable for learning and forecasting the given data. As the initial ANN structure is established before the pruning, we need to start from a fully connected topology, with one hidden layer, bias and direct shortcut connections, from input to output nodes. To enhance nonlinearity, the sigmoid activation function is applied on the hidden nodes. In the output node, the linear function is used to scale the range of the outputs, since the sigmoid function has a [0, 1] output range. This architecture is often adopted by the forecasting community and can avoid the need of filtering, which may give rise to information loss. Therefore, the general model is expressed as follows (Cortez et al., 2006):

$$x_t = w_{Out,0} + \sum_{i=1}^I x_{t-k_i} w_{Out,i} + \sum_{j=I+1}^{Out-1} f(\sum_{i=1}^I x_{t-k_i} w_{i,j} + w_{j,0}) w_{Out,j}, \quad (1)$$

where

- x_{t-k_i} is the past value in the time series as inputs,
- \wedge
- x_t the forecasted value of the time series at sequence number t as output,
- $w_{i,j}$ denotes the weight of the connection from node j to i (if $j = 0$ then it is a bias connection),
- “Out” denotes the output node,
- f is the sigmoid function $1 / (1 + e^{-x})$,
- K_i denotes the lag of the input i , and
- I is the number of input neurons.

THE EVOLUTIONARY MODEL

A critical issue for modeling time series by neural networks is the model selection. This involves the search of the suitable sliding time window and the ANN topology for the studied time series data. These selections govern the final forecasting performance of the model. Small time windows will provide insufficient information to the network, while a large time window may increase the entropy and negatively affect the learning. A network with few hidden neurons will have limited learning capabilities. On the other hand, using many hidden nodes will overfit the training data and introduce bias in the forecasting results. A statistical approach to model selection is to consider different candidate models, which are evaluated according to a generalization estimate. Several complex estimators have been developed, which are computationally burdensome. A reasonable alternative is to penalize model complexity by using a simple statistic. The Bayesian Information Criterion (*BIC*) is suggested by Cortez et al., (2006):

$$BIC = N \ln(\Delta / N) + P \ln(N),$$

where

N = the number of training cases,

P = the number of parameters, and

Δ = the total squared error which is defined as $\Delta = \sum_{\forall i} (Y_i - T_i)^2$ (Y_i denotes the forecasted value and T_i is the actual value of the future data point).

Although originally proposed for linear models, this *BIC* criterion has also been revised for neural estimation. When applied to neural networks, the number of parameters (P) is equal to the number of connection weights.

The *BIC* suggests small time windows and linear models (with no hidden node) for linear series. However, the *BIC* statistic also favors simple models (with zero or one hidden node) for the nonlinear series. Heuristic strategy can be employed in model selection, yet with some limitations. For example in terms of the *BIC* criterion, when adopting fully connected networks, the *BIC* prejudices networks with hidden nodes as shown by Cortez et al., (2006). Moreover, the sliding window rules are based on autocorrelation values, which only measure linear interactions, thus being inadequate for nonlinear

series. In the following, an evolutionary approach with the genetic algorithm is discussed in the model selection process to overcome these drawbacks.

The genetic algorithms (Holland 1973) are a particular class of evolutionary algorithms. There are a number of potential solutions (individuals) to a problem, evolving simultaneously (a population). Each individual is coded by a string (chromosome) of symbols (genes), taken from a well-defined alphabet. Each individual is assigned a numeric value (fitness) that indicates the solution's appropriateness. In each generation, a fraction of the individuals is replaced by the offsprings, which are generated by the application of genetic operators, such as crossover and mutation, in order to create new solutions (i.e., reproduction). The whole process is evolutionary, where the fittest individuals have greater chances of survival. The choice of initial population, the process of selection, cross and mutation are important factors which influence the stability and efficiency of genetic algorithms. We refer the readers to (Mitchell 1996) for more details.

The evolutionary optimization has been widely employed as the searching techniques to identify ANN topologies for forecasting tasks (Yao 1999; Dagli 1995). These models concentrated on the study of the fitness function of EC and the representation of the ANN. Regarding the fitness function, the most usual approach is to consider an error measure over an independent validation set. The representation scheme has been addressed by two alternatives, namely direct and indirect encodings. The direct encoding scheme encodes all the topology details (weights and connections), and it is the most frequently used approach (since it is easier and more efficient to implement). The indirect scheme makes use of the construction rules. The indirect encoding is more scalable and biologically plausible, favoring regular networks. However, since only modular networks are considered, the search space for the best topology is restricted and this approach has reported some drawbacks in many real-world problems (Siddiqi et al. 1998).

When designing multilayer perceptrons for time series forecasting, trial-and-error procedures, such as testing several combinations of hidden nodes, can be used (Faraway et al. 1998). The use of evolutionary design brings an alternative attempt for time series neural topology selection. Indeed, several evolutionary neural networks have been proposed, most of them use indirect parametric representation, and encode factors such as the number of input and hidden nodes, the initial weights, activation functions, or even learning rates (Falco et al. 1998; Chen et al. 1999). However, from a model selection point of view, adjusting the multilayer perceptron's connectivity seems to be more important. Cortez et al. (2006) employ an evolutionary neural network approach with a direct binary representation in time series forecasting. Since statistical analysis has shown that time series are often modeled by small networks, this encoding can prevent the topologies from growing too large. In fact, each gene represents a possible connection, if its value is 1, then the corresponding connection exists, otherwise it is not considered.

Assuming a maximum of I inputs and H hidden nodes, bias, and shortcut connections, the size of the chromosome is given by $(I + 1)(H + 1)$. Hidden node pruning will occur when there is no connection from input nodes, and input node pruning will occur when an input node has no output. This allows a complete representation, defining a search space which contains all neural topologies from the simplest linear perceptron to the fully connected multilayer network, and also to reach any subset of input nodes. In this way, only the starting numbers of H and I are needed to be set at the beginning of the search process, and they need not be encoded specifically. The final numbers of H and I can be retrieved from the final binary representation of the neural network structure. While pruning may occur during the evolution process, the final numbers can be smaller than the starting ones. The structure of neural network does

influence its performance, but the relation is indirect, changes of performance according to network structure are discrete and can provide discontinuous effects, where similar topologies may present different performances and distinct architectures may provide similar outputs. Thus, the encoding of H and I need not be done separately and the direct binary representation is a good option.

Regarding the fitness function, it is natural to take the performance criterion, such as the total squared error, from the validation test on the neural networks as the fitness function. A validation test is employed to avoid the problem of over-fitting (Zhang et al., 1998), and is extremely important for the case of having a small training set, like forecasting for the first few weeks of the sales pattern of fashionable products. The time series of the sales data have a strong “temporal” nature, e.g., a factor which influences the sales of a fashion item often lasts for just a few weeks (May-Plumlee et al., 2001). Data points from the most recent week are then proposed to be chosen as the validation dataset, so that the “time series in the week to forecast” and the “time series in the week for validation” are more likely to share a similar feature. As the *BIC* criterion presents the advantages of requiring a simple computation while increasing the selection pressure in favor of simpler structures, the *BIC* measure is used as the fitness function so that we can find the optimized networks with a simple structure.

AN OVERRFITTING TEST

This ENN approach described in previous section works fine when the maximum number of hidden nodes (H) is selected properly. However, there are cases where H is large and far greater than the required hidden neuron number in a network to provide optimized forecasting. Such a network can easily go overfit to the given time series and it is difficult to produce ideal forecasting. In this situation, it usually takes quite a long time for the approach to converge to a final result. Sometimes it cannot find the proper small structure network since the *BIC* criterion seems to prejudice networks with more hidden nodes if fully connected networks are adopted (Cortez et al., 2006). Apparently, the maximum neuron number H is a crucial parameter in this evolution approach. When H is large, this approach cannot always successfully identify a simple structure network for the forecasting even with the guide of *BIC*. Concerning the computational performance, the evolution process from a maximum 7 neurons network to a 2 neurons network can be very lengthy. To overcome these limitations, we introduce a pre-search algorithm to this model selection approach. It is obvious that when the neuron number in a network is too large, it will overfit a time series, the network will then fail to forecast the future of the time series correctly. In (Au et al., 2008), a pre-search mechanism is devised to find an appropriate maximum hidden neuron number H to start the evolution. By this approach, the evolutions process performance is expected to be improved.

Overfitting is one of the most critical issues in neural computation for forecasting, and many techniques have been developed (Bender, 1996; Geman et al., 1992). The objective of this algorithm is to determine the appropriate H , at which the evolution approach can quickly find two groups of individuals: One overfits the time series and one does not. A simple and intuitive method which decides the overfitting by observing the mean squared error and gradient in learning is employed. The algorithm starts from a given H . Since it takes more hidden neuron to overfit a time series with more data, the initial value of H is obtained from a preliminary test in which all datasets of a time series are fed in, since with hidden neuron H the network already overfits the whole sales time series, we take it as a starting value. Then, we follow the criteria window to adjust the value of H (e.g., if few overfitting cases are found, H is in-

Table 1. Parameters included in the overfitting test

Parameters	Meaning
α	Overfitting threshold
$[\underline{\alpha}, \bar{\alpha}]$	Test window
T	Upper bound on the testing time
K	Upper bound on the number of tests

creased; if many overfitting cases are found, H is decreased). The algorithm stops when the proportion of overfitting cases (y) is around a given value α where $0 < \underline{\alpha} < \alpha < \bar{\alpha} < 1$, i.e. $y \in [\underline{\alpha}, \bar{\alpha}]$. $[\underline{\alpha}, \bar{\alpha}]$ is known as the testing window. This algorithm is used to speed-up the whole evolution process. Since the overfitting test itself could be time-consuming, we have the following proposal to avoid it: a) We set a time limit T on the test. b) We impose a upper limit on the number of tests, called K . c) We set a loose criteria window. The loose criteria window is in favor of a greater H , and can guarantee that the optimal network structure can be found in the evolution process. Since the purpose of this algorithm is to find the starting H , so the overfitting threshold α need not be too rigid (in the literature, the value of α has been suggested to be 50% (see Au et al. 2008) even though this need not be optimal and we postpone the optimization on α to our future research). In order to guarantee that a safe H can be obtained in this process, the value of H is increased by 1 as the final result. Since starting from a slightly greater H can yield the same neural structure in the evolution process, the algorithm stops either when the running time of tests exceeds T or the number of tests exceeds K . When $[\underline{\alpha}, \bar{\alpha}]$ cases in the tests are found to be “overfitting”, the condition is met and the value of required H is found. This approach certainly may find a slightly greater H than is expected, but it is already much faster in running time than assigning a very big H which is safe to start with. In this way, the initial population which is generated with H hidden neurons is expected to give just sufficient individuals to overfit the time series. As H is just a starting number, and will be reduced in the evolution process, it is safe to find an H with which a fully connected network easily overfits the time series (i.e., it is considered big enough to start the evolution process). Overfitting of the present time series will be harmful to the forecasting of the future. With this starting H , the evolution process can be free of random searching for too many overfitting cases, and hence it can evolve to the optimal result faster. A summary of the required parameters are as shown in Table 1. As a remark, the parameters can be set with respect to the prior knowledge and preference of the decision maker regarding the performance of the algorithm (precision versus speed). An optimal selection on the values of these parameters will be explored in our future research.

EMPLOYING THE ENN MODEL IN FORECASTING

To illustrate the ENN forecasting method, sales data of three fashionable products are employed and studied. Tables 2, 3, and 4 illustrate the time series which consist of the number of goods sold per time point for three respective products I, II, III. From the real data set, it is interesting to observe that time series 1 and 3 exhibit global trends and seasonality. Time series 2 refers to another fashionable product and there is no obvious global trend; the fluctuation of its sales is random and persists all over its life

Table 2. Time series 1

Time	Sales (normalized)	Time	Sales (normalized)
1	0.026	11	0.083
2	0.052	12	0.083
3	0.07	13	0.153
4	0.048	14	0.17
5	0.026	15	0.066
6	0.074	16	0.079
7	0.074	17	0.074
8	0.066	.	.
9	0.044	.	.
10	0.114	.	.

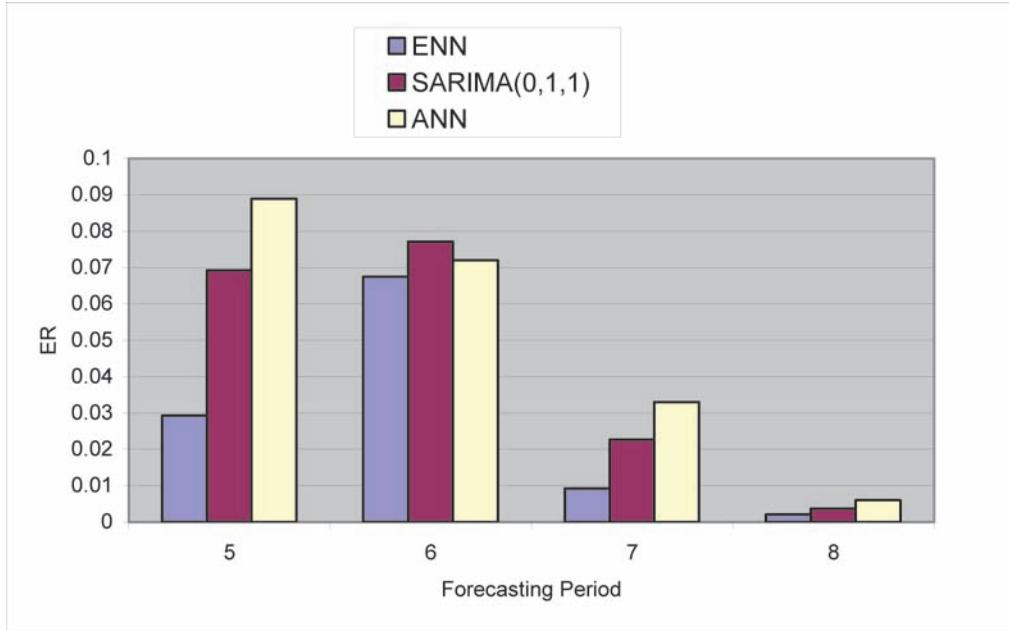
Table 3. Time series 2

Time	Sales (normalized)	Time	Sales (normalized)
1	0.325	11	0.286
2	0.39	12	0.545
3	0.597	13	0.558
4	1	14	0.779
5	0.649	15	0.429
6	0.74	16	0.39
7	0.844	17	0.364
8	0.338	.	.
9	0.299	.	.
10	0.61	.	.

Table 4. Time series 3

Time	Sales (normalized)	Time	Sales (normalized)
1	0.783	11	0.174
2	1	12	0.174
3	1	13	0.37
4	0.978	14	0.522
5	0.609	15	0.348
6	0.478	16	0.043
7	0.957	17	0.043
8	0.609	.	.
9	0.457	.	.
10	0.391	.	.

Figure 3. Comparisons among the forecasting models for time series 1



cycle. With these given sales data, we will proceed to conduct sales forecasting by using ENN, ANN and the traditional SARIMA with a parameter set of (1, 1, 0). In order to demonstrate the forecasting accuracy, we define the following.

The *mean squared error (ER)* is used for representing the forecasting accuracy in many parts of our study, it is given by:

$$ER = \Delta / \eta , \quad (2)$$

where

Δ is the total squared error,

η is the number of points in the sales time series.

In addition, we define the square root of ER , SER , as follows,

$$SER = \sqrt{ER} . \quad (3)$$

Notice that SER can give a more precise comparison when the ER is small.

After conducting the forecasting processes under ENN and the other models, Figure 3 presents a comparison among them for time series 1. As indicated in Figure 3, the ENN model outperforms both ANN and SARIMA. The performance of traditional fully-connected ANN is observed to be worse than that of the ENN in all cases, and only in Period 6, its forecasting accuracy is close to ENN and is better than SARIMA. This shows that an ANN network which overfits the time series would yield bad forecasting, which is a good support for making use of ENN instead of ANN for sales forecasting of

Table 5. Sample forecasted and actual data points under the ENN model

Time	Actual	Forecasted	Time	Actual	Forecasted
1	0.311688	0.5352	15	0.285714	0.3551
2	0.376623	0.4351	16	0.207792	0.2325
3	0.701299	0.4885	17	0.194805	0.4091
4	0.350649	0.5038	18	0.285714	0.2126
5	0.272727	0.5664	19	0.194805	0.1985
6	0.441558	0.5884	20	0.311688	0.2074
7	0.376623	0.5356	21	0.428571	0.6682
8	0.168831	0.1485	22	0.220779	0.2142
9	0.116883	0.0137	23	0.298701	0.2925
10	0.168831	0.0334	24	0.207792	0.2448
11	0.272727	0.256	25	0.324675	0.3042
12	0.181818	0.6254	26	0.207792	0.3116
13	0.415584	0.4419	27	0.324675	0.3545
14	0.649351	0.1663

fashionable products.

Further data analysis, based on the raw time series data and the forecasted results (see Table 5 for a sample extract of the data points) reveals that the ENN model produces better results when the variance in the time series is smaller and with less abnormality. Since the ENN structure is specifically simplified to adapt to the target time series, the nonlinear factors (hidden neurons and connections to them) are less significant and the linear factors (output neuron and connection to them) are more significant than the traditional ANN. Moreover, with these simple structures, it is difficult to produce fault tolerant feature. The ENN model inherits the computing burdensome of neural networks and it often takes a day to produce forecasting results, while the SARIMA model often completes the forecasting computation in seconds. Although ENN is relatively more time consuming than the SARIMA model, considering its better accuracy, it still has its strength in real applications. Figures 4 and 5 show the comparisons of forecasting performance of different methods with time series 2 and 3, and they show similar results as Figure 3's: ANN is generally the worst one among all the models, and although the SARIMA model outperforms the ENN in a small number of cases, ENN generally performs better than the others.

CONCLUSION

In this chapter, we have found that although fashion sales are highly volatile, the evolutionary neural network (ENN) can provide reasonably accurate forecasting. With the evolutionary searching approach, the non-fully connect neuron network is often found to be more accurate in forecasting for time series than the fully connect neuron network. Computation cost is one of the major drawbacks for neural network. However, when guided with *BIC* criterion and the pre-search approach, the Evolutionary Neural Network (ENN) can converge much faster. Forecasting is often time crucial, especially for fashion sales forecasting which often occurs weekly or even daily, the improvement of convergence speed makes this

Figure 4. Comparisons among the forecasting models for time series 2

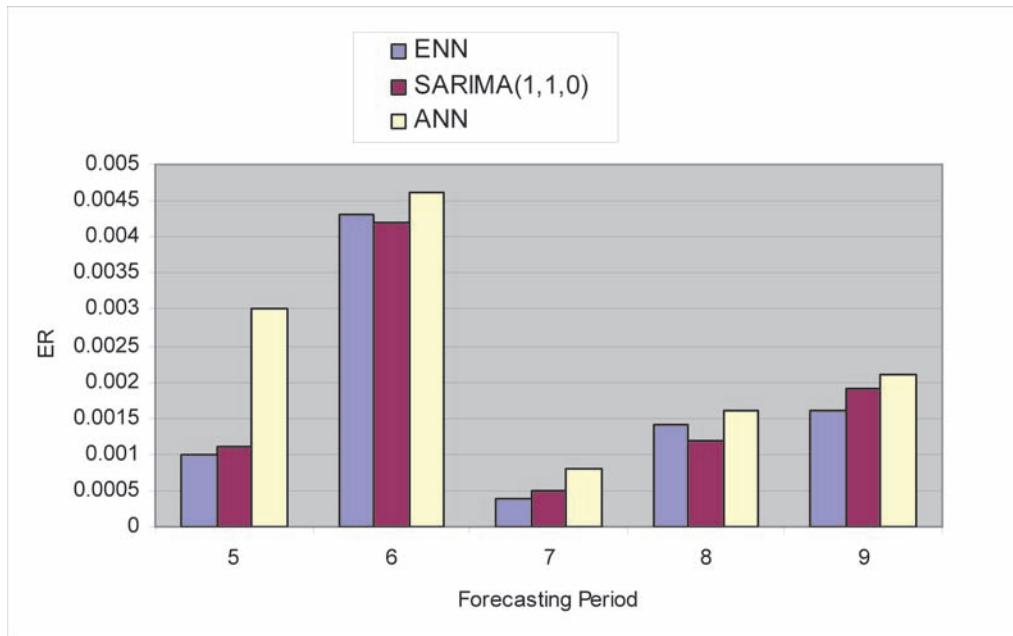
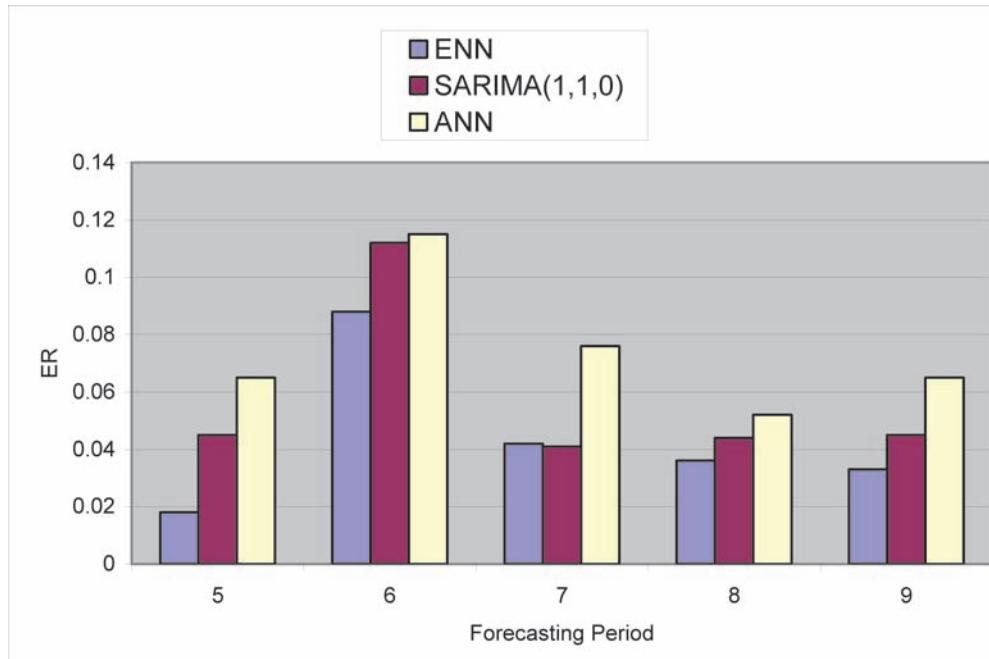


Figure 5. Comparisons among the forecasting models for time series 3



approach widely applicable to decision making problems such as fashion sales forecasting. Traditional forecasting approaches, like SARIMA, show their good performance in the forecasting for seasonal time series, but ENN can produce comparable forecasting results. Traditional fully-connected feed-forward ANN in forecasting is worse than other models in most cases because “overfitting” of the forecasting data is present, and this directly indicates the need for employing ENN. In fact, ENN performs very well and it can outperform the SARIMA model. Moreover, the ENN approach for forecasting is an almost automatic one while the SARIMA model involves more human knowledge, and this makes ENN a better and more convenient option for forecasting. While the ENN model gains accuracy with the evolution process, it also requires a much longer computation time. This burden is significantly alleviated by our proposed pre-search algorithm which makes the ENN method highly applicable and can assist managers to make scientific decisions in the sales forecasting of fashionable products. Future research on this topic can be conducted in the direction of analytically revealing the relationship between the forecasting accuracy and the features of the sales time series. Scientifically determining the optimal parameters in the forecasting algorithm is another interesting area for in-depth exploration.

ACKNOWLEDGMENT

We sincerely thank Professor Marcelino Martinez, the co-editor, for his kind invitation to contribute to this handbook. Thanks must also be given to the anonymous reviewers for their helpful comments. Kin-Fan Au's research is partially funded by Hong Kong Research Grant Council, GRF project account: PolyU 5101/05E. Tsan-Ming Choi's research is partially funded by the research grant provided by the Hong Kong Polytechnic University.

REFERENCES

- Abdi, H. (2003). Neural networks. In *Encyclopedia of social science research methods*. Thousand Oaks, CA: Sage.
- Amirikian, B., & Nishimura, H. (1994). What size network is good for generalization of a specific task of interest? *Neural Networks*, 7(2), 321–329. doi:10.1016/0893-6080(94)90026-4
- Au, K. F., Choi, T. M., & Yu, Y. (2008). Fashion retail forecasting by evolutionary neural networks. *International Journal of Production Economics*, 114, 615–630. doi:10.1016/j.ijpe.2007.06.013
- Baum, E. B., & Haussler, D. (1989). What size net gives valid generalization? *Neural Computation*, 1, 151–160. doi:10.1162/neco.1989.1.1.151
- Bender, E. A. (1996). *Mathematical methods in artificial intelligence*. Los Alamitos, CA: IEEE.
- Box, G. E. P., & Jenkins, G. M. (1970). *Time series analysis - forecasting and control*. San Francisco: Holden Day.
- Cavalieri, S., Maccarrone, P., & Pinto, R. (2004). Parametric vs. neural network models for the estimation of production costs: A case study in the automotive industry. *International Journal of Production Economics*, 91(2), 165–177. doi:10.1016/j.ijpe.2003.08.005

- Chatfield, C. (1993). Neural networks: Forecasting breakthrough or passing fad? *International Journal of Forecasting*, 9, 1–3. doi:10.1016/0169-2070(93)90043-M
- Chen, S., & Lu, C. (1999). Would evolutionary computation help in designs of ANNs in forecasting foreign exchange rates? In *Proceedings of the 1999 Congress on Evolutionary Computation, Vol. 1* (pp. 267-274).
- Choi, T. M. (2007). Pre-season stocking and pricing decisions for fashion retailers with multiple information updating. *International Journal of Production Economics*, 106, 146–170. doi:10.1016/j.ijpe.2006.05.009
- Chu, C. W., & Zhang, G. P. (2003). A comparative study of linear and nonlinear models for aggregate retail sales forecasting. *International Journal of Production Economics*, 86(3), 217–231. doi:10.1016/S0925-5273(03)00068-9
- Cortez, P., Rocha, M., Machado, J., & Neves, J. (1995). A neural network based forecasting system. In *Proceedings of the ICNN'95 – IEEE Int. Conf. on Neural Networks*, Perth, Western Australia (pp. 2689-2693).
- Cortez, P., Rocha, M., & Neves, J. (2006). Time series forecasting by evolutionary neural networks. In *Artificial neural networks in real-life applications* (pp. 47-70).
- Dagli, C. H., & Sittisathanchai, S. (1995). Genetic neuro-scheduler: A new approach for job shop scheduling. *International Journal of Production Economics*, 41(1-3), 135–145. doi:10.1016/0925-5273(95)00072-0
- De Toni, A., & Meneghetti, A. (2000). The production planning process for a network of firms in the textile-apparel industry. *International Journal of Production Economics*, 65, 17–32. doi:10.1016/S0925-5273(99)00087-0
- Dorffner, G. (1996). Neural networks for time series processing. *Neural Networks World*, 6(4), 447–468.
- Falco, I., Cioppa, A., Iazzetta, A., Natale, P., & Tar, E. (1998). Optimizing neural networks for time series prediction. In *Proceedings of the Third World Conference on Soft Computing, (WSC3)*.
- Faraway, J., & Chatfield, C. (1998). Time series forecasting with neural networks: A case study. *Applied Statistics*, 47, 231–250. doi:10.1111/1467-9876.00109
- Frank, C., Garg, A., Raheja, A., & Sztandera, L. (2003). Forecasting women's apparel sales using mathematical modeling. *International Journal of Clothing Science and Technology*, 15(2), 107–125. doi:10.1108/09556220310470097
- Fukushima, K. (1975). Cognitron: A self-organizing multilayered neural network. *Biological Cybernetics*, 20, 121–136. doi:10.1007/BF00342633
- Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1), 1–58. doi:10.1162/neco.1992.4.1.1

- Guo, Z., & Uhrig, R. (1992). Using genetic algorithm to select inputs for neural networks. In *Proceedings of the Workshop on Combinations of Genetic Algorithms and Neural Networks, COGANN92* (pp. 223-234).
- Hamilton, J. D. (1994). *Time series analysis: An applied focus with emphasis on economics and assumption reader has an economics background*. Princeton, NJ: Princeton U. Press.
- Hammond, J. H. (1990). *Quick response in the apparel industries*. Cambridge, MA: Harvard Business School.
- Hanke, J. E., Wichern, D. W., & Reitsch, A. G. (2001). *Business forecasting* (7th ed.). Upper Saddle River, NJ: Prentice Hall.
- Happel, B. L. M., & Murre, M. J. (1994). The design and evolution of modular neural network architectures. *Neural Networks*, 7, 985–1004. doi:10.1016/S0893-6080(05)80155-8
- Hill, T., Marquez, L., O'Connor, M., & Remus, W. (1994). Artificial neural network models for forecasting and decision making. *International Journal of Forecasting*, 10(1), 5–15. doi:10.1016/0169-2070(94)90045-0
- Holland, J. H. (1973). Genetic algorithms and the optimal allocation of trials. *SIAM Journal on Computing*, 2(2), 88–105. doi:10.1137/0202009
- Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2, 359–366. doi:10.1016/0893-6080(89)90020-8
- Huang, W., Xu, B., & Chan-Hilton, A. (2004). Forecasting flows in Apalachicola River using neural networks. *Hydrological Processes*, 18, 2545–2564. doi:10.1002/hyp.1492
- Jones, A. J. (1993). Genetic algorithms and their applications to the design of neural networks. *Neural Computing & Applications*, 1, 32–45. doi:10.1007/BF01411373
- Kuan, C. M., & Liu, T. (1995). Forecasting exchange rates using feedforward and recurrent neural networks. *Journal of Applied Econometrics*, 10(4), 347–364. doi:10.1002/jae.3950100403
- Kwok, T., & Yeung, D. (1997). Constructive algorithms for structure learning in feed forward neural networks for regression problems: A survey. *IEEE Transactions on Neural Networks*, 8(3), 630–645. doi:10.1109/72.572102
- Lapedes, A., & Farber, R. (1987). *Non-linear signal processing using neural networks: Prediction and system modeling* (Tech. Rep. LA-UR-87-2662). Los Alamos National Laboratory.
- Lee, H. C., & Dagli, C. H. (1997). A parallel genetic-neuro scheduler for job-shop scheduling problems. *International Journal of Production Economics*, 51(1-2), 115–122. doi:10.1016/S0925-5273(97)00073-X
- Luxhoj, J. T., Riis, J. O., & Stensballe, B. (1996). A hybrid econometric-neural network modeling approach for sales forecasting. *International Journal of Production Economics*, 43(2-3), 175–192. doi:10.1016/0925-5273(96)00039-4

- May-Plumlee, T., & Little, T. J. (2001). consumer purchase data as a strategic product development tool. *Journal of Textile and Apparel, Technology and Management*, 1(3), 1–10.
- McMullen, P. R. (2001). A Kohonen self-organizing map approach to addressing a multiple objective, mixed-model JIT sequencing problem. *International Journal of Production Economics*, 72(1), 59–71. doi:10.1016/S0925-5273(00)00091-8
- Miller, G. F., Todd, P. M., & Hegde, S. U. (1989). Designing neural networks using genetic algorithms. In J. D. Schaffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms* (pp. 370-384). San Francisco: Morgan Kaufman.
- Mitchell, M. (1996). *An introduction to genetic algorithms*. Cambridge, MA: MIT Press.
- Pao, Y. H. (1989). *Adaptive pattern recognition and neural networks*. Reading MA: Addison-Wesley.
- Poli, I., & Jones, R. D. (1994). A neural net model for prediction. *Journal of the American Statistical Association*, 89(425), 117–121. doi:10.2307/2291206
- Schiffmann, W., Joost, M., & Werner, R. (1993). Application of genetic algorithms to the construction of topologies for multi layer perceptron. In *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms* (pp. 675-682).
- Schikora, P. F., & Godfrey, M. R. (2003). Efficacy of end-user neural network and data mining software for predicting complex system performance. *International Journal of Production Economics*, 84(3), 231–253.
- Shtub, A., & Versano, R. (1999). Estimating the cost of steel pipe bending, a comparison between neural networks and regression analysis. *International Journal of Production Economics*, 62(3), 201–207. doi:10.1016/S0925-5273(98)00212-6
- Siddiqi, A., & Lucas, S. (1998). A comparison of matrix rewriting versus direct encoding for evolving neural networks. In *Proceedings of the IEEE Int. Conf. on Evolutionary Computation* (pp. 392-397).
- Sun, Z. L., Au, K. F., & Choi, T. M. (2007). A hybrid neuron-fuzzy inference system through integration of fuzzy logic and extreme learning machines. *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics*, 37(5), 1321–1331. doi:10.1109/TSMCB.2007.901375
- Sun, Z. L., Choi, T. M., Au, K. F., & Yu, Y. (2008). Sales forecasting using extreme learning machine with applications in fashion retailing. *Decision Support Systems*, 46, 411–419. doi:10.1016/j.dss.2008.07.009
- Sun, Z. L., Huang, D. S., Zheng, C. H., & Shang, L. (2006). Optimal selection of time lags for TDSEP based on genetic algorithm. *Neurocomputing*, 69(7-9), 884–887. doi:10.1016/j.neucom.2005.06.010
- Szstandera, L. M., Frank, C., & Vemulapali, B. (2004). Prediction of women's apparel sales using soft computing methods. In *Knowledge-based intelligent information and engineering systems* (LNCS 3214, pp. 506-512). Berlin, Germany: Springer.
- Tang, Z. Y., Almeida, C., & Fishwick, P. A. (1991). Time series forecasting using neural networks vs. Box- Jenkins methodology. *Simulation*, 57, 303–310. doi:10.1177/003754979105700508

- Thimm, G., & Fiesler, E. (1995). Evaluating pruning methods. In *Proc. of the International Symposium on Artificial Neural Networks* (pp. 20-25).
- Thomassey, S., Happiette, M., & Castelain, J. M. (2005). A global forecasting support system adapted to textile distribution. *International Journal of Production Economics*, 96(1), 81–95. doi:10.1016/j.ijpe.2004.03.001
- White, H. (1988). Economic prediction using neural networks: The case of IBM daily stock returns. In *Proceedings of the Second Annual IEEE Conference on Neural Networks, II* (pp. 451-458).
- Wu, P., Fang, S., King, R. E., & Nuttle, H. L. W. (1994). Decision surface modelling of apparel retail operations using neural network technology. In *Proc. of the IEEE 1994 Textile, Fiber and Film Industrial Conference*.
- Wu, P., Yang, W., & Wei, N. (2004). An electromagnetism algorithm of neural network analysis – an application to textile retail operation. *Journal of Chinese Institute of Industrial Engineers*, 21(1), 59–67.
- Yao, X. (1999). Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9), 1423–1447. doi:10.1109/5.784219
- Zhang, G., Patuwo, B. E., & Hu, M. Y. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, 14, 35–62. doi:10.1016/S0169-2070(97)00044-7

KEY TERMS AND DEFINITIONS

Artificial Neural Network (ANN): is a mathematical model or computational model based on biological neural networks. It consists of an interconnected group of artificial neurons and processes information using a connectionist approach to computation.

Bayesian Information Criterion (BIC): is a useful statistical criterion for model selection.

Evolutionary Computation (EC): is the optimization procedure which can mimic natural selection and biological evolution. Evolutionary Computation uses iterative progress, such as growth or development in a population. This population is then selected in a guided random search using parallel processing to achieve the desired end. Such processes are often inspired by biological mechanisms of evolution.

Evolutionary Neural Network (ENN): is the hybrid combination of evolutionary computation and neural networks.

Sales Forecasting: is a quantitative prediction of the future sales pattern. In this paper, this pattern is a time series.

SARIMA: Seasonal Autoregressive Integrated Moving Average Model is a time series analysis approach.

Time Series: a time series is a sequence of data points, measured typically at successive times, spaced at time intervals.

Chapter 19

Support Vector Machine based Hybrid Classifiers and Rule Extraction thereof: Application to Bankruptcy Prediction in Banks

M. A. H. Farquad

Institute for Development and Research in Banking Technology (IDRBT), INDIA University of Hyderabad, INDIA

V. Ravi

Institute for Development and Research in Banking Technology (IDRBT), INDIA

Raju S. Bapi

University of Hyderabad, INDIA

ABSTRACT

Support vector machines (SVMs) have proved to be a good alternative compared to other machine learning techniques specifically for classification problems. However just like artificial neural networks (ANN), SVMs are also black box in nature because of its inability to explain the knowledge learnt in the process of training, which is very crucial in some applications like medical diagnosis, security and bankruptcy prediction etc. In this chapter a novel hybrid approach for fuzzy rule extraction based on SVM is proposed. This approach handles rule-extraction as a learning task, which proceeds in two major steps. In the first step the authors use labeled training patterns to build an SVM model, which in turn yields the support vectors. In the second step extracted support vectors are used as input patterns to fuzzy rule based systems (FRBS) to generate fuzzy “if-then” rules. To study the effectiveness and validity of the extracted fuzzy rules, the hybrid SVM+FRBS is compared with other classification techniques like decision tree (DT), radial basis function network (RBF) and adaptive network based fuzzy inference system. To illustrate the effectiveness of the hybrid developed, the authors applied it to solve a bank bankruptcy prediction problem. The dataset used pertain to Spanish, Turkish and US banks. The quality of the extracted fuzzy rules is evaluated in terms of fidelity, coverage and comprehensibility.

INTRODUCTION

Support Vector Machines (SVMs) (Vapnik 1995, 1998) and other linear classifiers are popular methods for building hyperplane-based classifiers from data sets and have been shown to have excellent generalization performance in a variety of applications. SVM is based on the statistical learning theory developed by Vapnik (1995) and his team at AT&T Bell Labs, which is a new learning algorithm and can be seen as an alternative training technique for Polynomial, Radial Basis Function and Multi-Layer Perceptron classifiers (Cortes & Vapnik 1995, Edgar et al. 1997). The scheme used by the SVM is that some linear weighted sum of the explanatory variables is lower (or higher) than a pre-specified threshold indicating that the given sample is classified into one class or the other. Even though such a schemes works well, it is completely non-intuitive to human experts in that it does not let us know the knowledge learnt by it during training in simple, comprehensible and transparent way. Therefore, SVM are also treated as “Black Box” models just like ANN.

There are many techniques existing for extracting knowledge embedded in trained neural networks in the form of *if-then-else* rules (Tickle et al. 1998). The process of converting opaque models into transparent models is often called *Rule Extraction*. These models are useful for understanding the nature of the problem and interpreting its solution. Using the rules extracted one can certainly understand in a better way, how a prediction is made. Gallant (1988) initiated the work of rule extraction from a neural network that defines the knowledge learnt in the form of *if-then* rules.

The advantages of rule extraction algorithm:

- Provision of user explanation capability (Gallant 1988). Davis et al. (1977) argues that even limited explanation can positively influence the system acceptance by the user.
- Data exploration and the induction of scientific theories. A learning system might discover salient features in the input data whose importance was not previously recognized (Craven & Shavlik 1994).
- Improves Generalization.

A taxonomy describing the techniques that are used to extract symbolic rules from the neural networks is proposed by Andrew et al. (1995). In general, rule extraction techniques are divided into two major groups i.e. decompositional and pedagogical. Decompositional techniques view the model at its minimum (or finest) level of granularity (at the level of hidden and output units in case of ANN). Rules are first extracted at individual unit level, these subset of rules are then aggregated to form global relationship. Pedagogical techniques extract global relationship between the input and the output directly without analyzing the detailed characteristics of the underlying solution. The third group for rule extraction techniques is eclectic, which combines the advantages of the decompositional and pedagogical approaches.

Earlier work (Andrews et al. 1995), (Towell & Shavlik 1993) and recent work (Kurfess 2000, Darbari 2001) includes rule extraction from neural network. Hayashi (1990) incorporated fuzzy sets with expert systems and proposed a rule extraction algorithm FNES (Fuzzy Neural Expert System). FNES relies on the involvement of an expert at input phase for transforming the input data into required format. This transformation process is then automated in Fuzzy-MLP (Mitra 1994). Later Mitra & Hayashi (2000) presented a survey of extracting fuzzy rules from neural networks. However, not much work has been done to extract rules from SVM.

Researchers developed different algorithms to provide explanation capability to SVM. Fuzzy-support vector machine (Inoue 2001) is also developed to deal with pattern classification tasks. SVM+Prototype (Nunez et al. 2002) uses K-means clustering prototype vectors for each input class are determined, these prototypes are combined with support vectors to define an ellipsoid in the input space then mapped to *if-then* rules. The construction of ellipsoids with axes parallel to the coordinate axes, which reduces interpretability of the rules extracted, is the main drawback of this approach. RulExtSVM (Fu et al. 2004) extracts *if-then* rules using intervals defined by hyperrectangular forms, which is processed in three steps. First, a hyperrectangle is generated using the intersection of the support vectors with the SVM decision boundary. Second, the initial rule set is tuned in order to improve rule accuracy. In the final step the redundant rules have been removed to obtain more concise rule set. The disadvantage of this algorithm is the construction of as many hyperrectangles as the number of support vectors that can be computationally very expensive. Fung et al. (2005) developed an extracting algorithm, which extracts non-overlapping rules by constructing hyperplane with axis parallel surface. Barakat & Diederich (2004 & 2005) presented a hybrid technique for extracting rules, which first trains an SVM and removes the labels of the support vectors, then uses the developed model to predict the label for the training sample. Later, use these support vectors are used for training the decision tree and generating rules. Hyperrectangle Rules Extraction (HRE) (Zhang et al. 2005) first constructs hyperrectangles according to the prototypes and the support vectors (SVs) using SVM model. When these hyperrectangles are projected onto coordinate axes *if-then* rules are obtained.

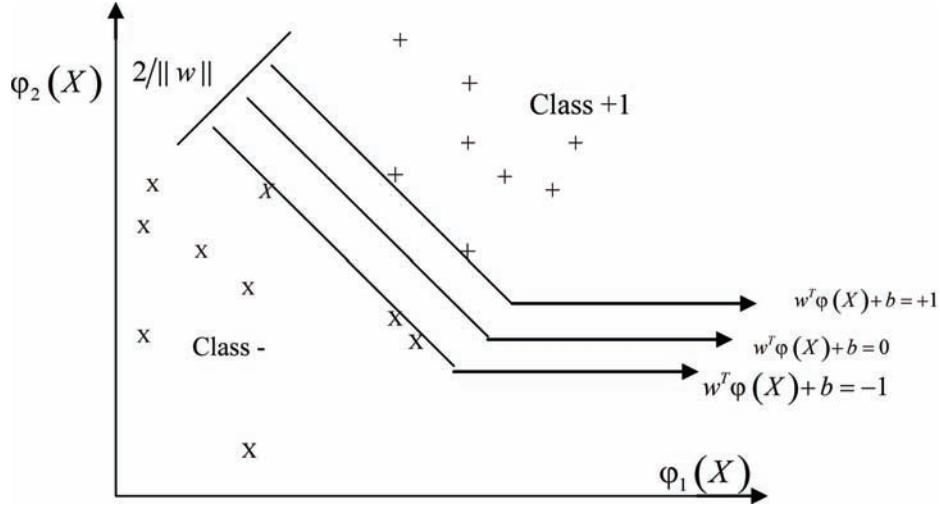
FREx (Chaves et al. 2005) Fuzzy Rule Extraction consists of three phases. During first phase projection of the support vectors in the coordinate axes is determined. In the second phase, given number of triangular fuzzy sets are constructed for each coordinate. Finally each support vector is transformed into *if-then* rule.

Four rule-extraction quality criteria were suggested in (Andrews et al. 1995): *rule accuracy, fidelity, consistency and comprehensibility*. In this context, a rule set is considered to be *accurate* if it can correctly classify previously unseen examples. Similarly a rule set is considered to display a high level of *fidelity* if it can mimic the behavior of the machine learning technique from which it was extracted. An extracted rule set is considered to be *consistent* if, under different training sessions the machine learning technique generates same rule sets that produce the same classifications of unseen examples. Finally, the *comprehensibility* of a rule set is determined by measuring the size of the rule set (in terms of number of rules) and the number of antecedents per rule.

It may be noted that the hybrid approach of Barakat & Diederich (2004 & 2005) yields Boolean rules. Further, their approach lacks clarity in moving over from support vectors to the decision tree classifier. Consequently, in this chapter, we propose a hybrid approach for fuzzy rule extraction from SVM by hybridizing it with FRBS. We then applied it to solve bankruptcy prediction in banks. This hybrid yields a compact set of fuzzy ‘if-then’ rules, which are more human comprehensible than their Boolean counterparts. Also, to test the effectiveness of the proposed SVM+FRBS hybrid, we compared its performance with that of SVM+DT, SVM+ANFIS and SVM+RBF hybrids.

The rest of the chapter is organized as follows. Section 2 describes overview of SVM and Fuzzy Rule Based System. Section 3 explains the proposed hybrid approach. Section 4 presents a brief literature review of bankruptcy prediction models. Section 5 presents the experimental setup. Section 6 presents results & discussion of the proposed approach. Finally, section 7 concludes the chapter.

Figure 1. Illustration of SVM optimization of the margin in the feature space



OVERVIEW OF SVM AND FUZZY RULE BASED SYSTEMS

Support Vector Machine

The support vector machine (SVM) is a universal constructive learning procedure based on the statistical learning theory (Vapnik 1995). SVMs are an inductive machine learning techniques based on the structural risk minimization principal that aims at minimizing the true error and performs classification by constructing an N -dimensional hyper plane that optimally separates the data into two categories. The main objective of SVM is to find an optimal separating hyperplane that correctly classifies data points as much as possible and separates the points of two classes as far as possible, by minimizing the risk of misclassifying the training samples and unseen test samples. SVM models are closely related to neural networks. In fact, a SVM model using a sigmoid kernel function is equivalent to a two-layer perceptron neural network.

Properties of Support Vector Machines:

- SVM provides the flexibility in choosing a similarity (distance) function.
- Solution can be achieved by using sparse training data i.e. a few samples are usually considered “important” by the algorithm. It is then crucial for SVM to keep number of support vectors as small as possible without compromising the accuracy of the classification.
- SVMs have the ability to handle large feature spaces.
- No probability density estimation is done.
- Perhaps the biggest limitation of the support vectors approach is choice of the kernel, speed and size.

SVM first formulates a constrained optimization problem and then solves it using Constrained Quadratic Programming (QP). Quadratic programming problems can be solved from its dual problem by introducing Lagrangian Multipliers (Lin & Wang 2002). Using the dot product functions in feature

space that is called kernels, SVM tries to find the optimal hyperplane for classification of the classes of two-class problem as shown in figure 1. The solution of the optimal hyperplane can be written as a combination of a few input points that are called support vectors.

Given a set of points $b \in \Re$ with $i=1\dots N$ each point x_i belongs to either of two classes with the label $y_i \in \{-1, +1\}$, Cristianini and Shawe-Taylor (2000).

The set S is linearly separable if there exist $w \in \Re^n$ and $b \in \Re$ such that,

$$\begin{cases} w^T \varphi(X_i) + b \geq +1, & y_i = +1, \\ w^T \varphi(X_i) + b \leq -1, & y_i = -1, \end{cases} \quad (1)$$

which is equivalent to

$$y_i [w^T \varphi(X_i) + b] \geq 1, \quad i = 1, \dots, N. \quad (2)$$

The nonlinear function $\varphi(\cdot)$ maps the input space to a high dimensional feature space. In this feature space, the above inequalities basically construct a hyperplane $w^T \varphi(X) + b = 0$ discriminating between both classes as in the fig. 1. for a typical two-dimensional case.

By minimizing $w^T w$, the margin between two classes is maximized.

The assumption in (2) was that such a function f actually exists that classifies all input pairs (x_i, y_i) or in other words that the convex optimization problem is feasible. Sometimes this may not be the case or to allow some errors i.e. soft margin loss function Cortes & Vapnik (1995) slack variables ξ_i are introduced.

In primal weight space the classifier then takes the form

$$y(X) = \text{sign}[w^T \varphi(X) + b], \quad (3)$$

Convex optimization problem is as follows

$$\min_{w,b,\xi} \vartheta(w, b, \varphi) = \frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i \quad (4)$$

subject to

$$\begin{cases} y_i [w^T \varphi(X_i) + b] \geq 1 - \xi_i & i = 1, \dots, N, \\ \xi_i \geq 0, & i = 1, \dots, N. \end{cases} \quad (5)$$

The first part of the objective function tries to maximize the margin between both classes in the feature space, whereas the second part minimizes the misclassification error. C is the positive real constant used as tuning parameter in the algorithm.

This problem is more or less tractable for real applications. In order to easily explain the extraction to nonlinear decision surfaces lagrangian multipliers are used and lagrangian is constructed (Osuna et al. 1997).

The lagrangian to the constraint optimization problem (4) and (5) is given by

$$L(w, b, \xi; \alpha, v) = \vartheta(w, b, \xi) - \sum_{i=1}^N \alpha_i \left\{ y_i [w^T \varphi(X_i) + b] - 1 + \xi_i \right\} - \sum_{i=1}^N v_i \xi_i \quad (6)$$

The saddle point of the Lagrangian gives the solution to the optimization problem, i.e. By minimizing $L(w, b, \xi; \alpha, v)$ with respect to w, b, ξ and maximizing it with respect to α and v .

This leads to the following classifier:

$$y(X) = \text{sign} \left[\sum_{i=1}^N \alpha_i y_i K(X_i, X) + b \right], \quad (7)$$

whereby $K(X_i, X) = \varphi(X_i)^T \varphi(X)$ is taken with a positive definite kernel satisfying the Mercer theorem (Boser et al. 1992). The Lagrange multipliers α_i are then determined by means of the following optimization problem (dual problem):

$$\max_{\alpha_i} -\frac{1}{2} \sum_{ij=1}^N y_i y_j K(X_i, X_j) \alpha_i \alpha_j + \sum_{i=1}^N \alpha_i \quad (8)$$

subject to

$$\begin{cases} \sum_{i=1}^N \alpha_i y_i = 0, \\ 0 \leq \alpha_i \leq C, \quad i = 1, \dots, N \end{cases} \quad (9)$$

The entire classifier construction problem now simplifies to a convex quadratic programming (QP) problem. It's not mandatory to calculate w or $\varphi(X_i)$ in order to determine the decision surface. Thus, no explicit construction of the linear mapping $\varphi(X)$ is needed. Instead of this, the kernel function $K(., .)$ will be used.

For kernel function $K(., .)$ one typically has the following choices:

$$K(X, X_i) = X_i^T X, \quad (\text{Linear Kernel})$$

$$K(X, X_i) = (1 + X_i^T X / C)^d, \quad (\text{Polynomial kernel of degree } d)$$

$$K(X, X_i) = \exp \left\{ - \|X - X_i\|_2^2 / \sigma^2 \right\}, \quad (\text{RBF kernel})$$

$$K(X, X_i) = \tanh(\kappa X_i^T X + \theta), \quad (\text{MLP kernel})$$

where d, c, σ, κ and θ are constants.

For low-noise problems, many of the α_i will be typically equal to zero (sparseness property). The training observations corresponding to non-zero α_i are called support vectors and are located close to the decision boundary.

Fuzzy Rule Based System

Fuzzy logic is a generalization of conventional (Boolean) logic that has been extended to handle the concept of partial truth and was introduced by Lotfi Zadeh (1965) at the University of California, Berkeley. Zadeh expanded traditional set theory by making membership in a set i.e. a matter of degree rather than a ‘yes’ or ‘no’ situation. Fuzzy logic includes ‘0 and 1’ as the extreme cases of truth (fact) but also include the various states of truth in between 0 and 1. So that for example the result of a comparison between two things could not be “Tall” or “Short” but “0.45 of Tall”. Fuzzy Logic incorporates a simple, rule-based *IF X AND Y THEN Z* approach towards solving a control problem rather than attempting to model a system mathematically.

The compactness of the rules is desirable because there exists evidence suggesting smaller rules that perform better (Holte 1993), with reasons essentially the same as those for over fitting in decision trees. Fuzzy logic appears to be very well suited for the creation of small rules, as fuzzy rules have a higher ‘information density’ i.e. each rule encapsulates a richness of information and meaning.

Rule: IF a person is a "heavy_smoker"
THEN the risk of cancer is "high"

where the two fuzzy concepts “*heavy_smoker*” and “*high*” can be represented by their membership function values.

Fuzzy rule-based systems have been successfully applied to various control problems (Sugeno 1985, Lee 1990). Fuzzy grid with pre-specified membership function is preferred when the objective is comprehensibility of fuzzy-rules by the user. Ishibuchi et al. (1992) proposed a heuristic method for generating fuzzy if-then rules for pattern classification problems using grid based fuzzy partition and pre-specified membership functions. Rule selection methods based on genetic algorithms were proposed in (Ishibuchi et al. 1995 and 1997) (1997). Genetic algorithms have been widely used for generating fuzzy if-then rules and tuning membership functions (Carse et al. 1996).

Ishibuchi et al. (1999) examined the performance of a fuzzy genetic algorithm - based machine learning method for multidimensional pattern classification problem with continuous attributes, where each fuzzy if-then rule is handled as an individual and a fitness value is assigned to each rule. It is observed that grid based fuzzy partitions cannot handle high-dimensional data i.e. when we use the grid-type fuzzy partition with the increase in comprehensibility the number of fuzzy if-then rules increases exponentially as the number of input variables increase. *Don't care* antecedent is introduced to deal with the curse of dimensionality and to produce less number of fuzzy if-then rules by genetic operations. Antecedent *don't care* is represented by an interval-type membership function whose membership value is always unity in the domain of each attribute value. For example, if the domain of the i^{th} attribute (i.e. x_i) is the unit interval $[0, 1]$, the membership function of the antecedent fuzzy set “*don't care*” can be written as

$$\mu_{don't-care} = \begin{cases} 1, & \text{if } 0 \leq x_i \leq 1, \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

Rule Generation

In Ishibuchi (1999) they used fuzzy if then rules of the following type of c -class pattern classification problem with the n -dimensional pattern space $[0, 1]^n$.

Rule R_j : If x_1 is A_{j1} and ... and x_n is A_{jn}
 Then Class C_j with $CF = CF_j$ (11)

where

R_j Label of the j^{th} fuzzy if-then rule;
 $A_{j1} A_{jn}$ Antecedents fuzzy sets $[0, 1]$;
 C_j Consequent class;
 CF_j Grade of certainty of the fuzzy if-then rule R_j .

Grade of certainty CF_j is different from the fitness value of each rule. The fitness value is used in a selection operation of fuzzy classifier system while CF_j is used in fuzzy reasoning for classifying new patterns. If we use three linguistic values for rules i.e. for the n -dimensional pattern classification problem, the total number of if-then rules is $(3+1)^n$. It is very difficult to use all $(3+1)^n$ fuzzy rules in fuzzy rule-based systems. This fuzzy classifier system searches for a set of relatively small number of fuzzy if-then rules using genetic operators.

The consequent class C_j and the grade of certainty CF_j of each fuzzy if then rule are determined using;

Step1: Calculate the compatibility grade of each training pattern $x_p = (x_{p1}, x_{p2}, \dots, x_{pn})$ with fuzzy if-then rule R_j using the product operation:

$$\mu(x_p) = \mu_{j1}(x_{p1}) \times \dots \times \mu_{jn}(x_{pn}) \quad (12)$$

where $\mu_{ji}(x_{pi})$ is the membership function of A_{ji}

Step2: sum of the compatibility grades for each class with the fuzzy if-then rule R_j

$$\beta_{Class-h}(R_j) = \sum_{x_p \in Class-h} \mu_j(x_p), h=1, 2, \dots, c. \quad (13)$$

where $\beta_{Class-h}(R_j)$ is the sum of compatibility grades of training pattern in class h with rule Rj.

Step3: find Class \hat{h}_j that has the maximum value of $\beta_{Class-h}(R_j)$

$$\beta_{Class-\bar{h}_j}(R_j) = \text{Max}\{\beta_{Class-1}(R_j), \dots, \beta_{Class-c}(R_j)\}. \quad (14)$$

If two or more classes take the maximum value, the consequent class C_j of to fuzzy if-then rule R_j can not be determined uniquely. In this case, let C_j be \emptyset . If a single class takes the maximum value, let C_j be Class \hat{h}_j . If there is no training pattern compatible with fuzzy if-then rules R_j (i.e. if $\beta_{Class-h}(R_j) = 0$ for $h = 1, 2, \dots, c$), the consequent class C_j is also specified as \emptyset .

Step4: if the consequent class C_j is \emptyset , let the grade of certainty CF_j of the fuzzy if-then rule R_j be $CF_j = 0$. Otherwise the grade of certainty CF_j is determined as follows:

$$CF_j = \{\beta_{Class-\bar{h}}(R_j) - \bar{\beta}\} / \sum_{h=1}^c \beta_{Class-h}(R_j) \quad (15)$$

$$\text{where } \bar{\beta} = \sum_{\substack{h=1 \\ h \neq \bar{h}_j}} \beta_{Class-h}(R_j) / (c-1). \quad (16)$$

The above procedure for two-class classification problem will become

$$CF_j = \frac{\beta_{Class1}(R_j) - \beta_{Class2}(R_j)}{\beta_{Class1}(R_j) + \beta_{Class2}(R_j)}. \quad (17)$$

in this case $\beta_{Class1}(R_j) = \beta_{Class2}(R_j)$, C_j is \emptyset and $CF_j = 0$ because it cannot specify the consequent class C_j .

The winner rule R_j for the input pattern x_p is determined as

$$\mu_{\bar{j}}(x_p) \cdot CF_{\bar{j}} = \text{Max}\{\mu_j(x_p) \cdot CF_j \mid R_j \in S\}. \quad (18)$$

that is the winner rule has the maximum product of the compatibility $\mu_j(x_p)$ and the grade of certainty CF_j .

If more than one rule has the same maximum product but different consequent classes for the input pattern x_p , the classification of that pattern is rejected. Classification also rejected if no fuzzy if-then rule is compatible with the input pattern x_p (i.e. $\mu_j(x_p) = 0$ for $\forall R_j \in S$).

A unit is awarded to the winner rule when a training pattern is correctly classified by that rule. After all the training patterns are examined, the fitness value of each fuzzy if-then rule is defined as follows by the total reward assigned to that rule.

$$\text{fitness}(R_j) = NCP(R_j) \quad (19)$$

where $\text{fitness}(R_j)$ is the fitness value of the fuzzy if-then rule R_j , and $NCP(R_j)$ is the number of training patterns that are correctly classified by R_j .

Outline of fuzzy classifier system (Ishibuchi et al. 1999) is as follows

- Step1:** Generate an initial population of N_{pop} fuzzy if-then rules by randomly specifying antecedent fuzzy sets of each rule. The consequent class and the grade of certainty are determined by the heuristic procedure explained in section rule generation.
- Step2:** Classify all the given training patterns by the fuzzy if-then rules in the current population and then calculate the fitness value of each rule by (19).
- Step3:** Generate N_{rep} fuzzy if-then rules from the current population by the selection, crossover and mutation operations. The consequent class and the grade certainty of each fuzzy if-then rule are determined by the heuristic procedure explained in rule generation section.
- Step4:** Replace the worst N_{rep} fuzzy if-then rules with the smallest fitness values in the current population with the newly generated rules.
- Step5:** Terminate the iteration of the algorithm if the pre-specified stopping condition is satisfied, otherwise return to Step2.

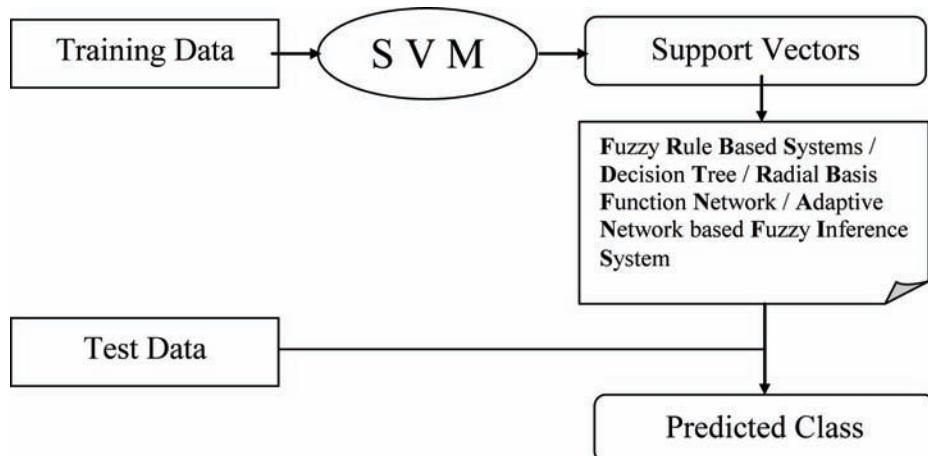
PROPOSED HYBRID APPROACH

In this research work we propose a novel hybrid approach for bankruptcy prediction in banks using SVM and FRBS, leading to a significant improvement in prediction accuracy. The interpretability of SVM is improved by fuzzy *if-then* rules. The hybrid is carried out in two steps; (i) support vectors are extracted using SVM and (ii) they are used as new training set for FRBS for fuzzy rule extraction. The classifier is then tested with test data. In order to evaluate the effectiveness and validity of the SVM+FRBS hybrid, we compared it with other hybrids such as SVM+DT, SVM+ANFIS and SVM+RBF. The hybrid SVM+RBF is chosen as a base line hybrid even though we don't get any rules out of it. The block diagram of the proposed hybrid classifiers is shown in figure 2.

LITERATURE REVIEW OF BANKRUPTCY PREDICTION IN BANKS AND FIRMS

Since 1960s (Altman 1968) bankruptcy prediction especially in banks has been extensively researched. Creditors, auditors, stockholders and senior management are all interested in bankruptcy prediction because

Figure 2. Work flow diagram of the Hybrid Approach



it adversely affects all of them (Wilson and Sharda, 1994). In the online methods on site examinations are conducted on banks premised by regulatory every 12-18 months as mandated by the federal deposit insurance corporation improvement act of 1991. Regulators utilizes six part rating system referred as CAMELS (Cole and Gunther, 1995), which evaluates banks according to their basic functional areas, *Capital adequacy, Assets quality, management expertise, Earnings, strength, Liquidity, and Sensitivity* to market risks.

Altman (1968) initiated the work of using financial ratios and multiple discriminant analysis to predict financially distressed firms. However, in general the usage of MDA relies on the restrictive assumptions on linear separability, multivariate normality and independence of the predictive variables (Kerals and Prakash, 1987, Ohlson, 1980). Bankruptcy prediction problem for financial firms can also be solved using wide range of classifier. Tam (1991) explored a back propagation neural network for this problem and compared its performance with another methods such as MDA, linear regression, K-NN and ID3. He concluded that the neural network outperformed other techniques. Salchenberger (1992) found that neural network produced fewer or equal number of classification error for each of the forecast periods compared to the logit model.

Consequently many researchers viewed neural network as an alternative to statistical techniques for bankruptcy prediction. Atiya (2001) provided a survey of all the prediction techniques including neural network applied to the bankruptcy prediction problem. He proposed a new neural network model using more financial indicators in addition to the traditional ones. Becerra et al. (2005) reported that wavelet neural network may have advantage over Multi layer perceptron and nonlinear models may be valid alternative to the linear discriminant models. Shin et al. (2005) applied Support Vector Machine to the problem of corporate bankruptcy prediction and concluded that SVM outperformed MLFF-BP in terms of accuracy, as the training dataset get smaller. Tsakonas et al. (2006) proposed a hybrid intelligent approach for bankruptcy prediction, combining neural logic network and genetic programming. The proposed approach is compared with rough set approach, discriminant analysis model and logit model.

Ravi Kumar and Ravi (2006a) reported that fuzzy rule based classifier outperformed the well-known technique MLFF-BP in case of US data. Ravi Kumar and Ravi (2006b) proposed an ensemble classifier, developed using simple majority voting scheme for bankruptcy prediction.

In another work, Ravi Kumar and Ravi (2007) conducted a comprehensive review of all the works reported for bankruptcy prediction in banks and firms during 1968-2005. It compares the results in terms of prediction accuracy, data sources, and timeline of each study wherever available. Hu and Tseng (2007) proposed a method for bankruptcy prediction called functional link network with fuzzy integrals (FIFLN), which is used to enhance the performance of the traditional single layer perceptron. They also designed a genetic algorithm based training method to estimate the weights in their network. They concluded that FIFLN outperformed the functional-link net and Multilayer Perceptron (MLP).

Pramodh and Ravi (2007) proposed and implemented a variant for Baek and Cho's neural network and named it Modified Great Deluge Algorithm based Auto Associative Neural Network (MGDAANN), wherein a metaheuristic is used to train the auto associative neural network they applied it to solve bankruptcy prediction in banks.

Later Ravi et al. (2007) proposed a semi-online training algorithm for Radial Basis Function Neural Network (SORBFN) applied to bankruptcy prediction and they concluded that SORBFN without linear terms performed better than techniques such as ANFIS, SVM, MLFF-BP, RBF and Orthogonal RBF. Ravi et al. (2008) developed a novel soft computing system fro bank performance prediction based on MLP, RBF, Classification and Regression Trees (CART), Probabilistic Neural Networks (PNN), Fuzzy

Rule Based Classifier (FRBC), and Principal Component Analysis (PCA) based hybrid techniques.

Karthikchandra and Ravi (2008) presented a novel hybrid intelligent system in the framework of soft computing to predict the failure of dotcom companies. The hybrid intelligent system comprises the techniques such as a MLP, Random Forest (RF), Logistic Regression (LR), SVM, Classification and Regression Trees (CART). Their results are superior to those reported in previous studies on the same data set. Ravi and Pramodh (2008) proposed an application of new principal component neural network (PCNN) architecture to bankruptcy prediction problem in commercial banks. It is inferred that the proposed PCNN hybrids outperformed other classifiers in terms of AUC. It is also observed that the proposed feature subset selection algorithm is very stable and powerful.

Rada (2008) presented a detailed review report regarding expert systems and evolutionary computing for financial investments. Queries like what changes have occurred over time in the tools used for financial applications, what financial application get how much attention and are knowledge based and evolution based methods being integrated, are explored basically. Most recently, Nikunj et al. (2008) proposed differential evolution algorithm for training the wavelet neural network for bankruptcy prediction in banks to improve the classification accuracy.

EXPERIMENT SETUP

Dataset is first split into two parts 80% and 20% respectively for training and validation purposes. 10 fold cross validation is carried out on the training data. For each fold, support vectors are extracted from training set of the fold and used as new training set for the classifiers FRBS, DT & ANFIS. The classifiers are tested using the test set of each fold. Whichever fold gives the best accuracy on the test set of the fold is selected and using the parameter combination employed for that fold, the classifier is in turn evaluated against validation set.

In order to extract support vectors from the SVM we used the function libSVM implemented in YALE 3.3 (2006). Later, fuzzy rules are generated using FRBS (Ishibuchi 1999) implemented in the data-mining tool KEEL 1.0.1 (2009). Then, finally these fuzzy rules are tested on validation set and classification accuracy, sensitivity and specificity of the hybrid model are reported. YALE 3.3 (2006) (<http://yale.sf.net>) is an open-source knowledge discovery environment. On the other hand, Keel 1.0.1 (Alcalá et al. 2009) is a software tool to assess evolutionary algorithms for data mining problems.

We used three well-known bank bankruptcy datasets to carry out experiments viz. (i) Spanish Banks (ii) Turkish Banks and (iii) US Banks. Information regarding the bankruptcy datasets is presented in Table 1.

Table 1. Spanish, Turkish and USA banks dataset details

Bank name	Total Instances	Attributes	Bankrupt	Non Bankrupt	Training 80%	Validation 20%
Spanish	66	9	29	37	53	13
Turkish	40	12	18	22	32	8
USA	130	5	66	64	104	26

RESULTS AND DISCUSSIONS

Turkish banks' dataset is obtained from (Canbas et al. 2005), which is available at (<http://www.tbb.org.tr/english/bulton/yillik/2000/ratios.xls>). Banks association of Turkey published 49 financial ratios. Initially, Canbas applied univariate analysis of variance (ANOVA) test on these 49 ratios of previous year for predicting the health of the bank in present year. However, Canbas & Kilic (2005) chose only 12 ratios as the early warning indicators that have the discriminating ability (i.e. significant level is $<5\%$) for healthy and failed banks one year in advance. Among these variables, 12th variable has some missing values meaning that the data for some of the banks are not given. So, we filled those missing values with the mean value of the variable following the general approach in data mining. This dataset contains 40 banks where 22 banks went bankrupt and 18 banks are healthy.

The Spanish banks' data is obtained from (Olmeda and Fernandez 1997). Spanish banking industry suffered the worst crisis during 1977-85 resulting in a total cost of 12 billion dollars. The ratios used for the failed banks were taken from the last financial statements before the bankruptcy was declared and the data of non-failed banks was taken from 1982 statements. This dataset contains 66 banks where 37 went bankrupt and 29 healthy banks.

The US banks' dataset contains 129 banks from the Moody's Industrial Manual, where banks went bankrupt during 1975-1982 (Rahimian et al. 1996). This dataset includes 65 bankrupt banks data and 64 healthy banks data.

The results of Spanish banks, Turkish banks and US banks are presented in Tables 3, 4, 5, 6, 7, and 8 with sensitivity and specificity presented. Sensitivity is the measure of proportion of the true positives which are correctly identified. Specificity is the measure of proportion of the true negatives, which are correctly identified. We also compared the results of the hybrid classifiers with decision tree J48, RBF and ANFIS in stand-alone mode. The financial ratios of the banks, which are used as explanatory variables, are presented in Table 2.

Results of Spanish Banks dataset presented in Table 3, results indicate that our hybrid method SVM+FRBS yielded 92.31% classification accuracy with 83.33% sensitivity and 100% specificity. RBF, DT, FRBS and ANFIS stand-alone classifiers gave 92.31%, 84.62%, 69.23% and 92.31% accuracy respectively with 83.33%, 66.67%, 33.33% and 100% sensitivity respectively. SVM+RBF resulted in 92.31% classification accuracy with 100% sensitivity and 85.71% specificity. SVM+DT and SVM+ANFIS also yielded 92.31% classification accuracy but 83.33% sensitivity and 100% specificity. It is observed that ANFIS stand-alone yielded 92.31% accuracy with 100% sensitivity and 85.71% specificity, while the hybrid SVM+ANFIS yielded 92.31% classification accuracy with 83.33% sensitivity and 100% specificity. SVM + FRBS also yielded 92.31% classification accuracy with 83.33% sensitivity and 100% specificity, which is comparable to stand-alone FRBS. Fuzzy rules set extracted using SVM+FRBS approach is presented in Table 4.

Table 5 presents the results of Turkish Banks dataset. RBF, SVM+RBF, ANFIS, SVM + ANFIS yielded 100% classification accuracy, sensitivity and specificity. DT stand-alone produced 75% accuracy with 50% sensitivity and 100% specificity, while SVM+DT improve the classification accuracy by yielding 87.5% with 75% sensitivity and 100% specificity. FRBS classifier yielded 75% classification accuracy, sensitivity and specificity, while SVM+FRBS resulted 87.5% accuracy with 100% sensitivity and 75% specificity. Fuzzy rules extracted for Turkish bank dataset using SVM+FRBS hybrid is tabulated in Table 6.

Table 2. Financial Ratios of Turkish, Spanish and USA banks data

Turkish banks' data	
#	Predictor Variable Name
1	<i>InterestEpenses/AverageProfitableAssets</i>
2	<i>InterestExpenses/AverageNon-ProfitableAssets</i>
3	<i>(Share Holders'Equity +TotalIncome)/(Deposits +Non-DepositFunds)</i>
4	<i>InterestIncome/InterestExpenses</i>
5	<i>(Share Holders'Equity +TotalIncome)/TotalAssets</i>
6	<i>(Share Holders'Equity +TotalIncome)/(TotalAssets +Contingencies &Commitments)</i>
7	<i>NetworkingCapital/TotalAssets</i>
8	<i>(Salary AndEmployees'Benefits +Reserve ForRetirement)/No. OfPersonnel</i>
9	<i>LiquidAssets/(Deposits +Non-DepositFunds)</i>
10	<i>InterestExpenses/TotalExpenses</i>
11	<i>LiquidAssets/TotalAssets</i>
12	<i>StandardCapitalRatio</i>
Spanish banks' data	
1	<i>CurrentAssets/TotalAssets</i>
2	<i>CurrentAssets-Cash/TotalAssets</i>
3	<i>CurrentAssets/Loans</i>
4	<i>Reserves/Loans</i>
5	<i>NetIncome/TotalAssets</i>
6	<i>NetIncome/TotalEquityCapital</i>
7	<i>NetIncome/Loans</i>
8	<i>Cost OfSales/Sales</i>
9	<i>CashFlow/Loans</i>
US banks' data	
1	<i>WorkingCapital/TotalAssets</i>
2	<i>RetainedEarnings/TotalAssets</i>
3	<i>Earnings BeforeInterest AndTaxes/TotalAssets</i>
4	<i>Market Value OfEquity/TotalAssets</i>
5	<i>Sales/TotalAssets</i>

Results of US banks dataset is presented in Table 7. Proposed hybrid SVM+FRBS yielded good classification accuracy of 96.15% with 92.31% sensitivity and 100% specificity. While RBF stand alone obtained 100% accuracy. Fuzzy rules extracted using US banks dataset are presented in Table 8. SVM+RBF resulted in 92.31% classification accuracy with 92.31% sensitivity and specificity. DT and SVM+DT achieved 92.31% classification accuracy with 100% sensitivity and 84.62% specificity. ANFIS stand-alone resulted in 92.31% accuracy; sensitivity and specificity. SVM+ANFIS yielded less accuracy of 88.46% with 84.62% sensitivity and 92.31% specificity. FRBS yielded 92.31% classification accuracy with 84.62% sensitivity and 100% specificity.

Table 3. Spanish bank results with validation set as test dataset

Experiments	Classification Accuracy %	Sensitivity %	Specificity %
RBF	92.31	83.33	100
S V + RBF	92.31	100	85.71
Decision Tree	84.62	66.67	100
SVM + Decision Tree	92.31	83.33	100
ANFIS	92.31	100	85.71
SVM + ANFIS	92.31	83.33	100
Fuzzy Rule Base System	69.23	33.33	100
SVM + FRBS	92.31	83.33	100

Table 4. Fuzzy rules extracted using Spanish banks dataset

Rule#	Antecedents	Consequent	Certainty Degree
01	If “R/L is Medium” and “CF/L is Medium” then	Non-Bankrupt	0.343
02	If “CAC/TA is Medium” and “R/L is Low” and “CF/L is Medium”	Bankrupt	0.516
03	If “R/L is Medium” and “CF/L is Low”	Bankrupt	0.483
04	If “R/L is Low”	Bankrupt	0.296
05	If “CA/TA is Medium” and “R/L is Low”	Bankrupt	0.389
06	If “CF/L is Low”	Bankrupt	0.289
07	If “CA/TA is Medium” and “R/L is Medium” and “CF/L is Medium”	Non-Bankrupt	0.469
08	If “R/L is High”	Non-Bankrupt	0.478
09	If “CS/S is High”	Bankrupt	1.0
10	If “CA/TA is High” and “CF/L is Medium”	Non-Bankrupt	0.755
11	If “R/L is High” and “CF/L is High”	Non-Bankrupt	1.0
12	If “R/L is Low” and “CF/L is Medium”	Bankrupt	0.421
13	If “R/L is Medium” and “CF/L is High”	Non-Bankrupt	0.759
14	If “CA/TA is Medium” and “CAC/TA is Medium” and “R/L is Medium” and “CF/L is Medium”	Non-Bankrupt	0.595

Table 5. Turkish banks results with validation set as test dataset

Experiment	Classification Accuracy %	Sensitivity %	Specificity %
RBF	100	100	100
SVM + RBF	100	100	100
Decision Tree	75	50	100
SVM + Decision Tree	87.5	75	100
ANFIS	100	100	100
SVM + ANFIS	100	100	100
Fuzzy Rule Base System	75	75	75
SVM + FRBS	87.5	100	75

Table 6. Fuzzy rules extracted using Turkish banks dataset

Rule#	Antecedents	Consequent	Certainty Degree
01	If “NC/TA is Low”	Bankrupt	1.0
02	If “SCR is Low”	Bankrupt	1.0
03	If “IE/APA is Low”	Non-Bankrupt	0.500
04	If “IE/ANA is Medium”	Bankrupt	0.415

Table 7. US banks results with validation set as test dataset

Experiment	Classification Accuracy %	Sensitivity %	Specificity %
RBF	100	100	100
SVM + RBF	92.31	92.31	92.31
Decision Tree	92.31	100	84.62
SVM + Decision Tree	92.31	100	84.62
ANFIS	92.31	92.31	92.31
SVM + ANFIS	88.46	84.62	92.31
Fuzzy Rule Base System	92.31	84.62	100
SVM + FRBS	96.15	92.31	100

Table 8. Fuzzy rules extracted using US banks dataset

Rule#	Antecedents	Consequent	Certainty Degree
01	If “RE/TA is High” and “MVE/TA is Medium”	Non-Bankrupt	0.483
02	If “RE/TA is High”	Non-Bankrupt	0.258
03	If “EIT/TA is High”	Non-Bankrupt	0.391
04	If “RE/TA is Low”	Bankrupt	0.913
05	If “RE/TA is High” and “EIT/TA is High”	Non-Bankrupt	0.672
06	If “RE/TA is Medium” and “EIT/TA is Medium” and “MVE/TA is Low”	Bankrupt	0.321
07	If “RE/TA is Medium” and “EIT/TA is Medium” and “S/TA is Medium”	Bankrupt	0.394
08	If “EIT/TA is Medium” and “S/TA is Medium”	Bankrupt	0.248
09	If “RE/TA is High” and “S/TA is Low”	Non-Bankrupt	0.367
10	If “WC/TA is Medium” and “EIT/TA is Low”	Bankrupt	0.862
11	If “EIT/TA is Low” and “S/TA is Medium”	Bankrupt	0.884

A Receiver Operating Characteristics (ROC) graph (Fawcett 2006) has long been used in signal detection theory to depict the tradeoff between hit accuracies and false alarm accuracies of classifiers. The ROC graph is a two dimensional graph which represents various classifiers based on their output results in the point form in a region, which has FP rate (1-Specificity) on the X-axis and TP rate (Sensitivity) on the Y-axis. They are able to provide a richer measure of classification performance than scalar measures such as accuracy, error rate or error cost. Higher the areas under ROC better the classifier figure 3. The AUC of a Classifier A can be calculated as the sum of the areas of Triangle-CEA, Rectangle-EAFI and

Figure 3. A U C of two classifiers A and B

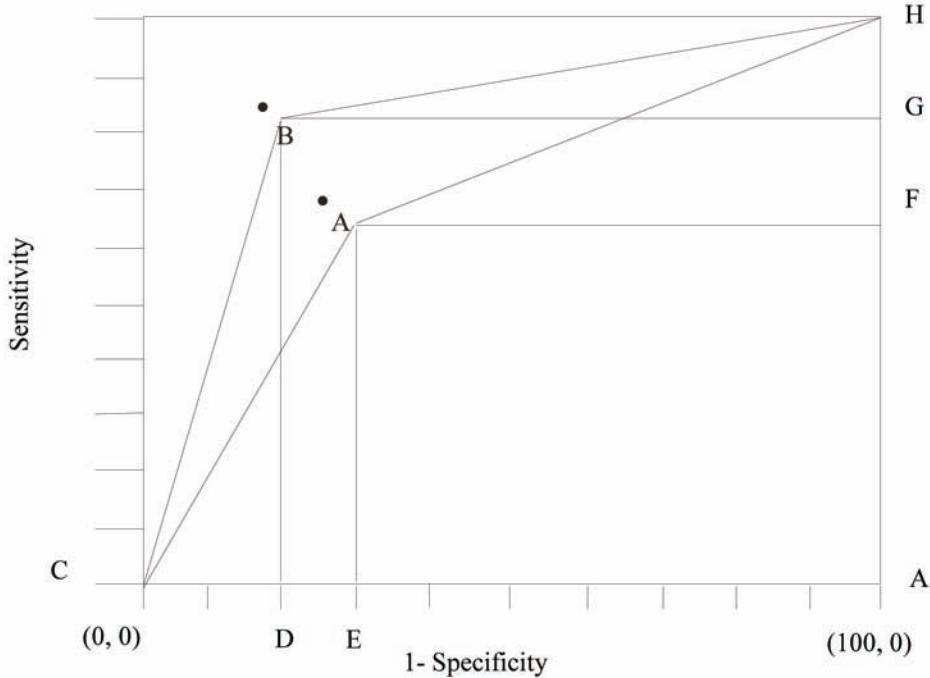


Table 9. A U C (Area under ROC (Receiver Operating Characteristic) curve)

	Spanish Banks	Turkish Banks	USA Banks
DT	8333.5	7500	9231
SVM + DT	9166.5	8750	9231
RBF	9166.5	10000	10000
SVM + RBF	9385.5	10000	9231
ANFIS	9385.5	10000	9231
SVM + ANFIS	9166.5	10000	8846
FRBS	6666.5	7500	9231
SVM + FRBS	9166.5	8750	9615.5

Triangle RAH. AUC (Area Under the ROC Curve) is calculated for all the classifiers and the hybrids are presented in Table 9.

Spanish Banks dataset shows ANFIS stand-alone and SVM + RBF classifiers hold the top position and are best classifiers covering the largest area under curve i.e. 9385.5. While classifiers SVM + DT, RBF, SVM + ANFIS, FRBS and SVM + FRBS have the area covered is 9166.5. DT stand-alone covers less area under curve i.e. 8333.5. Classifiers RBF, SVM + RBF, ANFIS and SVM + ANFIS cover the whole area under curve and are the best classifiers with respect to Turkish Banks dataset i.e. 10000. DT stand-alone classifier covers the area 7500 under curve and SVM + DT covering 8750 under curve

Table 10. Fidelity of various hybrids

	Spanish Banks	Turkish Banks	USA Banks
SVM + DT	83.04	90.24	85.71
SVM + RBF	78.1	89.21	86.98
SVM + ANFIS	77.85	85.21	73.06
SVM + FRBS	83	91.81	92.31

which is considerably more than stand-alone DT classifier. SVM + FRBS cover 8750 under curve which seems to be better than stand-alone FRBS 7500. The best classifier covering much of the area under curve using USA banks dataset is SVM + FRBS 9615.5, but RBF stand-alone covers the whole area under curve. DT, SVM + DT, SVM + RBF, ANFIS and FRBS classifiers fall on the same point on the AUC plot with the area 9231. Classifier SVM + ANFIS cover least area under curve 8846 and can be considered the worst classifier in the group.

A rule set is considered to display a high level of fidelity if it can mimic the behavior of the machine learning technique from which it was extracted. Our proposed approach is having high fidelity compared to other classifiers tested and the results are presented in Table 10. With Spanish Banks dataset SVM+FRBS shows 83% fidelity while other classifier SVM+DT, SVM+RBF and SVM+ANFIS have 83%, 78.1% and 77.85 respectively. Classifier SVM+FRBS using Turkish Banks dataset gives high fidelity compared with other classifiers i.e. 91.81%, other classifiers like SVM+DT, SVM+RBF and SVM+ANFIS yielded less fidelity of 90.24%, 89.21% and 85.21% respectively. Fidelity with respect to US banks data is high for SVM+FRBS classifier with fidelity 92.31%, while other classifiers yielded less fidelity compared with SVM+FRBS. SVM+DT, SVM+RBF and SVM+ANFIS yielded 85.71%, 86.98% and 73.06% respectively.

CONCLUSION

It is well known that ANN and SVM are black boxes in nature, meaning that they do not explicitly convey the knowledge learnt by them during training. Rule extraction techniques aim to remove this disadvantage by bringing in comprehensibility and interpretability to these stand-alone models. This chapter presents a novel hybrid approach for extracting fuzzy if-then rules by invoking SVM and FRBS in tandem. In order to test the validity and the effectiveness of the proposed hybrid, we tested it against other classifiers like DT, ANFIS and RBF and hybrids such as SVM+DT, SVM+ANFIS and SVM+RBF. We used three well-known bank bankruptcy datasets viz. Spanish banks, Turkish banks and US banks to carry out the experiments. The results indicate that the proposed hybrid SVM+FRBS shows superior performance using Spanish and US bank datasets. While for Turkish bank dataset SVM+RBF surpasses other classifiers in terms of accuracy. It is concluded that hybrid classifier yielded better classification accuracy and comprehensible fuzzy rules compared to stand alone classifiers.

REFERENCES

- Alcalá-Fdez, J., Sánchez, L., García, S., Del Jesus, M. J., Ventura, S., & Garrell, J. M. (2009). KEEL: A software tool to assess evolutionary algorithms for data mining problems. *Soft Computing*, 13(3), 307–318. doi:10.1007/s00500-008-0323-y
- Altman, E. (1968). Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *The Journal of Finance*, 23, 589–609. doi:10.2307/2978933
- Andrews, R., Diederich, J., & Tickle, A. B. (1995). Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, 8(6), 373–389. doi:10.1016/0950-7051(96)81920-4
- Atiya, A. F. (2001). Bankruptcy prediction for credit risk using neural networks: A survey and new results. *IEEE Transactions on Neural Networks*, 12, 929–935. doi:10.1109/72.935101
- Barakat, N., & Diederich, J. (2004). Learning-based rule-extraction from support vector machines. In *Proceedings of the 14th International Conference on Computer Theory and applications ICCTA'2004*, Alexandria, Egypt.
- Barakat, N., & Diederich, J. (2005). Eclectic rule-extraction from support vector machines . *International Journal of Computational Intelligence*, 2(1), 59–62.
- Becerra, V. M., Galvao, H., & Abou-Seads, M. (2005). Neural and wavelet network models for financial distress classification. *Data Mining and Knowledge Discovery*, 11, 35–55. doi:10.1007/s10618-005-1360-0
- Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In D. Haussler (Ed.), *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory* (pp. 144-152).
- Canbas, S. C., & Kilic, S. B. (2005). Prediction of commercial bank failure via multivariate statistical analysis of financial structures: The Turkish case. *European Journal of Operational Research*, 166, 528–546. doi:10.1016/j.ejor.2004.03.023
- Carse, B., Fogarty, T. C., & Munro, A. (1996). Evolving fuzzy rule based controllers using genetic algorithms. *Fuzzy Sets and Systems*, 80, 273–293. doi:10.1016/0165-0114(95)00196-4
- Chaves, A. C. F., Vellasco, M. M. B. R., & Tanscheit, R. (2005). *Fuzzy rule extraction from support vector machines*. Paper presented at the Fifth International Conference on Hybrid Intelligent Systems.
- Cole, R., & Gunther, J. (1995). A CAMEL rating's shelf life. *Federal Reserve Bank of Dallas Review*, December, 13-20.
- Cortes, C., & Vapnik, V. (1995). Support vector networks. *Machine Learning*, 20, 273–297.
- Craven, M. W., & Shavlik, J. W. (1994). Using sampling and queries to extract rules from trained neural networks. In *Machine Learning: Proceedings of the Eleventh International Conference*, San Francisco, CA, USA.

- Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. New York: Cambridge University Press.
- Darbari, A. (2001). *Rule extraction from trained ANN: A survey (research index)*. Institute of Artificial Intelligence, Dept. of Computer Science, TU Dresden, Germany.
- Davis, R., Buchanan, B. G., & Shortliffe, E. (1977). Production rules as a representation for a knowledge-based consultation program. *Artificial Intelligence*, 8(1), 15–45. doi:10.1016/0004-3702(77)90003-0
- Diederich, J., & Barakat, N. (2004). *Hybrid rule-extraction from support vector machines*. In *Proceedings of the IEEE conference on cybernetics and intelligent systems*, Singapore (pp. 1270-1275).
- Edgar, E. O., Freund, R., & Girosi, F. (1997). *Support vector machines: Training and applications* (Research report). Massachusetts Institute of Technology.
- Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27, 861–874. doi:10.1016/j.patrec.2005.10.010
- Fu, X., Ong, C. J., Keerthi, S., Hung, G. G., & Goh, L. (2004). Extracting the knowledge embedded in support vector machines. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'04)*, Budapest.
- Fung, G., Sandilya, S., & Bharat, R. R. (2005). Rule extraction from linear support vector machines. In *KDD '05: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining* (pp. 32-40). New York: ACM Press.
- Gallant, S. (1988). Connectionist expert systems. *Communications of the ACM*, 31(2), 152–169. doi:10.1145/42372.42377
- Hayashi, Y., & Imura, A. (1990). Fuzzy neural expert system with automated extraction of fuzzy If-then rules from a trained neural network. In *Proceedings of First International Symposium on Uncertainty Modeling and Analysis* (pp. 489-494).
- Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11, 63–91. doi:10.1023/A:1022631118932
- Hu, Y. C., & Tseng, F. M. (2007). Functional-link net with fuzzy integral for bankruptcy prediction. *Neurocomputing*, 70, 2959–2968. doi:10.1016/j.neucom.2006.10.111
- Inoue, T., & Abe, S. (2001). *Fuzzy Support vector machines for pattern classification*. In *Proceedings of the IJCNN'01, International Joint Conference on Neural Network*, 2 (pp. 1449-1454).
- Ishibuchi, H., Murata, T., & Turksen, I. B. (1997). Single-objective and two-objective genetic algorithms for selecting linguistics rules for pattern classification problems. *Fuzzy Sets and Systems*, 89, 135–149. doi:10.1016/S0165-0114(96)00098-X
- Ishibuchi, H., Nakashima, T., & Murata, T. (1999). Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 29(5), 601–618. doi:10.1109/3477.790443

- Ishibuchi, H., Nozaki, K., & Tanaka, H. (1992). Distributed representation of fuzzy rules and its application to pattern classification. *Fuzzy Sets and Systems*, 52, 21–32. doi:10.1016/0165-0114(92)90032-Y
- Ishibuchi, H., Nozaki, K., Yamamoto, N., & Tanaka, H. (1995). Selecting fuzzy if-then rules for classification problem using genetic algorithms. *IEEE transactions on Fuzzy Systems*, 3, 260–270. doi:10.1109/91.413232
- Karels, G. V., & Prakash, A. J. (1987). Multivariate normality and forecasting for business bankruptcy. *Journal of Business Finance & Accounting*, 14, 573–593. doi:10.1111/j.1468-5957.1987.tb00113.x
- Karthikchandra, D., Ravi, V., & Bose, I. (in press). Failure prediction of dotcom companies using hybrid intelligent techniques. *Expert Systems with Applications*. doi:.doi:10.1016/j.eswa.2008.05.047
- KEEL. (2009). *KEEL: Knowledge extraction based on evolutionary learning*. Retrieved from <http://sci2s.ugr.es/keel/index.php>
- Kurfess, F. J. (2000). Neural networks and structured knowledge: Rule extraction and applications. *Applied Intelligence*, 12(1/2), 7–13. doi:10.1023/A:1008344602888
- Lee, C. C. (1990). Fuzzy logic in control systems: Fuzzy logic controller, part I and part II. *IEEE Transactions on Systems, Man, and Cybernetics*, 20, 404–435. doi:10.1109/21.52551
- Lin, C.-F., & Wang, S.-D. (2002). Fuzzy support vector machines. *IEEE Transactions on Neural Networks*, 13(2).
- Mitra, S. (1994). Fuzzy MLP based expert system for medical diagnosis. *Fuzzy Sets and Systems*, 65, 285–296. doi:10.1016/0165-0114(94)90025-6
- Mitra, S., & Hayashi, Y. (2000). Neuro-fuzzy rule generation: Survey in soft computing framework. *IEEE Transactions on Neural Networks*, 11(3), 748–768. doi:10.1109/72.846746
- Nikunj, C., Ravi, V., & Karthik, C. (in press). Differential evolution trained wavelet neural networks: Application to bankruptcy prediction in banks. *Expert Systems with Applications*. doi:.doi:10.1016/j.eswa.2008.09.019
- Nunez, H., Angulo, C., & Catata, A. (2002). Rule extraction from support vector machines. In *Proceedings of the European Symposium on Artificial Neural Networks* (pp. 107-112).
- Ohlson, J. A. (1980). Financial ratios and the probabilistic prediction of bankruptcy. *Journal of Accounting Research*, 18, 109–131. doi:10.2307/2490395
- Olmeda, I., & Fernandez, E. (1997). Hybrid classifiers for financial multicriteria decision making: the case of bankruptcy prediction. *Computational Economics*, 10, 317–335. doi:10.1023/A:1008668718837
- Osuna, E. E., Freund, R., & Girosi, F. (1997). *Support vector machines: Training and applications* (Tech. Rep. AIM-1602).
- Pramodh, C., & Ravi, V. (2007). Modified great deluge algorithm based auto associative neural network for bankruptcy prediction in banks. *International Journal of Computational Intelligence Research*, 3(4), 363–370.

- Rada, R. (2008). Expert systems and evolutionary computing for financial investing: A review. *Expert Systems with Applications*, 34, 2232–2240. doi:10.1016/j.eswa.2007.05.012
- Rahimian, E., Singh, S., Thammachote, T., & Virmani, R. (1996). Bankruptcy prediction by neural network. In R. R. Trippi & E. Turban (Eds.), *Neural networks in finance and investing*. Burr Ridge, USA: Irwin Professional Publishing.
- Rapid-I. (n.d.). *Rapid-I*. Retrieved from <http://yale.sf.net>
- Ravi, V., Kurniawan, H., Thai, P. N., & Ravi Kumar, P. (2008). Soft computing system for bank performance prediction. *Applied Soft Computing Journal*, 8(1), 305–315. doi:10.1016/j.asoc.2007.02.001
- Ravi, V., & Pramodh, C. (2008). Threshold accepting trained principal component neural network and feature subset selection: Application to bankruptcy prediction in banks. *Applied Soft Computing*, 8(4), 1539–1548. doi:10.1016/j.asoc.2007.12.003
- Ravi, V., Ravi Kumar, P., Srinivas, E. R., & Kasabov, N. K. (2007). A semi-online training algorithm for the radial basis function neural networks: Applications to bankruptcy prediction in banks. In V. Ravi (Ed.), *Advances in banking technology and management: Impact of ICT and CRM*. Hershey, PA: IGI Global.
- Ravi Kumar, P., & Ravi, V. (2006). Bankruptcy prediction in banks by fuzzy rule based classifier. In *Proceedings of the 1st IEEE International Conference on Digital and Information Management*, Bangalore (pp. 222-227).
- Ravi Kumar, P., & Ravi, V. (2006a). Bankruptcy prediction in banks by an ensemble classifier. In *Proceedings of the IEEE International Conference on Industrial Technology*, Mumbai (pp. 2032-2036).
- Ravi Kumar, P., & Ravi, V. (2007b). Bankruptcy prediction in banks and firms via statistical and intelligent techniques - a review. *European Journal of Operational Research*, 180(1), 1–28. doi:10.1016/j.ejor.2006.08.043
- Salchenberger, L. C., & Lash, N. (1992). Neural networks, mine: A tool for predicting thrift failures. *Decision Sciences*, 23, 899–916. doi:10.1111/j.1540-5915.1992.tb00425.x
- Shin, K. S., Lee, T. S., & Kim, H. J. (2005). An application of support vector machines in bankruptcy prediction model. *Expert Systems with Applications*, 28, 127–135. doi:10.1016/j.eswa.2004.08.009
- Sugeno, M. (1985). An introductory survey of fuzzy control. *Information Sciences*, 36(1-2), 59–83. doi:10.1016/0020-0255(85)90026-X
- Tam, K. Y. (1991). Neural network models and the prediction of bank bankruptcy. *Omega*, 19, 429–445. doi:10.1016/0305-0483(91)90060-7
- Tickle, A. B., Andrews, R., Golea, M., & Diederich, J. (1998). The truth will come to light: Directions and challenges in extracting the knowledge embedded within trained artificial neural network. *IEEE Transactions on Neural Networks*, 9(6), 1057–1068. doi:10.1109/72.728352
- Towell, G., & Shavlik, J. (1993). The extraction of refined rules from knowledge based neural network. *Machine Learning*, 131, 71–101.

Tsakonas, A., Dounias, G., Doumpos, M., & Zopounidis, C. (2006). Bankruptcy prediction with neural logic networks by means of grammar-guided genetic programming. *Expert Systems with Applications*, 30, 449–461. doi:10.1016/j.eswa.2005.10.009

Vapnik, V. N. (1995). *The nature of statistical learning theory*. New York: Springer-Verlag.

Vapnik, V. N. (1998). *Statistical learning theory*. New York: John Wiley & Sons.

Wilson, R. L., & Sharda, R. (1994). Bankruptcy prediction using neural networks. *Decision Support Systems*, 11, 545–557. doi:10.1016/0167-9236(94)90024-8

Zadeh, L.A. (1965). Fuzzy sets. *Information and Control*, 8, 338–353. doi:10.1016/S0019-9958(65)90241-X

Zhang, Y., Su, H., Jia, T., & Chu, J. (2005). Rule extraction from trained support vector machines. In *Advances in knowledge discovery and data mining* (LNCS 3518, pp. 61–70). Berlin, Germany: Springer.

KEY TERMS AND DEFINITIONS

Adaptive Network-based Fuzzy Inference Systems (ANFIS): Using a given input/output data set: the toolbox function ANFIS constructs a fuzzy inference system (FIS) whose membership function parameters are tuned (adjusted) using either a backpropagation algorithm alone, or in combination with a least squares type of method

Decision Tree (DT): A “decide-and-conquer” approach to the problem of learning from a set of independent instances leads naturally to a style of representation called a decision tree.:

Fidelity: a rule set is considered to display a high level of fidelity if it can mimic the behavior of the machine learning technique from which it was extracted.:

Fuzzy Rule Based Systems: Fuzzy rules are linguistic IF-THEN- constructions that have the general form “IF A THEN B” where A and B are (collections of) propositions containing linguistic variables. A is called the premise and B is the consequence of the rule.:

Radial Basis Function Network (RBF): RBF is a feed-forward neural network and has both unsupervised and supervised phases. In the unsupervised phase: input data are clustered and cluster details are sent to hidden neurons, where radial basis functions of the inputs are computed by making use of the center and the standard deviation of the clusters

Rule Extraction: Rule extraction is the procedure to represent the knowledge in the form of IF-THEN rules: learnt by the model during training

Support Vector Machine (SVM): The SVM is a powerful learning algorithm based on recent advances in statistical learning theory. SVMs are learning systems that use a hypothesis space of linear functions in a high dimensional space: trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory

Chapter 20

Data Mining Experiences in Steel Industry

Joaquín Ordieres-Meré

Universidad Politécnica de Madrid, Spain

Ana González-Marcos

Universidad de La Rioja, Spain

Manuel Castejón-Limas

Universidad de León, Spain

Francisco J. Martínez-de-Pisón

Universidad de La Rioja, Spain

ABSTRACT

This chapter reports five experiences in successfully applying different data mining techniques in a hot-dip galvanizing line. Engineers working in steelmaking have traditionally built mathematical models either for their processes or products using classical techniques. Their need to continuously cut costs down while increasing productivity and product quality is now pushing the industry into using data mining techniques so as to gain deeper insights into their manufacturing processes. The authors' work was aimed at extracting hidden knowledge from massive data bases in order to improve the existing control systems. The results obtained, though small at first glance, lead to huge savings at such high volume production environment. The effective solutions provided by the use of data mining techniques along these projects encourages the authors to continue applying this data driven approach to frequent hard-to-solve problems in the steel industry.

INTRODUCTION

Iron and steel making industries rely on the quality of their products for survival. In their ceaseless need of quality improvement, engineers are more than happy to accept any helping hand the new technologies might lend to this otherwise mature and traditional activity. Engineers have traditionally used classical techniques to develop mathematical models to predict the behaviour of a product or process. Now,

DOI: 10.4018/978-1-60566-766-9.ch020

they are turning their eyes to data mining techniques so as to gain deeper insights on the manufacturing processes in order to develop strategies to cut costs down, to improve the quality and to increase the productivity.

The data mining approach is becoming increasingly popular because of two concurrent factors that seem to work in tandem. Firstly, the capacity to record data from the industrial processes has experienced a fast growth in the last years. As a consequence, day by day we have more and more information on the manufacturing processes stored in databases. ISO 9000 quality principles recommend adopting a factual approach to decision making, and having a database of the real facts is surely amongst the best approaches to accord with ISO principles. Nevertheless, as the data sets grow in size, the information overload might make more and more difficult to extract useful conclusions. Thus, the use of data mining tools capable of handling such massive data sets becomes mandatory. Secondly, the relationships amongst variables from industrial processes are, most frequently, non linear and thus explicit models might be hard to obtain. This compels the practitioner to based on data models as opposed to analytical models based on explicit equations. Fortunately, thanks to the development of increasingly faster computers this based on data approach is nowadays a feasible solution to otherwise difficult to solve frequent problems in industry.

This chapter reports some of the results obtained by the EDMANS (Engineering Data Mining And Numerical Simulations) group in optimizing a hot dip galvanising line at one of the ArcelorMittal spanish factories (Avilés). Several data mining techniques were successfully used in these projects, e.g. neural networks, expert systems and decision trees, in order to improve the control systems, to extract hidden knowledge and, thus, obtain improvements that mean huge savings in economic terms due to the large production volume.

BACKGROUND

Galvanized steel is a product experiencing a growing demand in sectors like the automotive industry, domestic appliances manufacturing and construction industry, due to its anticorrosive properties. As the requirements of the clients are increasingly harder to meet, companies pursuing to lead the market need to follow a continual improvement strategy covering every stage of the galvanizing process (Tian, Hou & Gao, 2000; Fabrellas, Ruiz, Martínez, Sanz, & Larrazábal, 2002; Díaz, Cuadrado, Diez, Rodríguez, Obeso & Rodríguez, 2003; Rendueles, González, Díaz, Diez, Seijo & Cuadrado, 2006). Two are the key aspects that determine the quality of the product (Schiefer, Jörgl, Rubenzucker & Aberl, 1999; Tenner, Linkens, Morris & Bailey, 2001; Domínguez, Miranda-Ariz & Salvador, 2002):

- The anticorrosive properties conferred by the zinc coating. The protection against corrosion requires that the zinc layer has uniform thickness. A strict control of the base metal surface preparation, the temperature, composition and homogeneity of the coating, the speed of the band and the air flow throw the blades is mandatory.
- The mechanical properties. They basically depend on the chemical composition of the steel, the rolling process and the thermal treatments before immersion in hot liquid zinc.

The five works reported here are focused on improving both the anticorrosive and mechanical properties. First, we tackle the problem of measuring the mechanical properties of galvanized steel coils.

Current sensing techniques only allow the measurement of the mechanical properties by destructive testing in the laboratory. Thus, the process is typically operated following an open loop actuation scheme. An artificial neural network based model was developed to obtain estimates of these properties in real time (Ordieres-Meré, González-Marcos, González & Lobato-Rubio, 2004). This greatly improved the responsiveness against departures from the desired quality. Second, we show how to improve the adhesion of the zinc coating by improving the estimates of the steel temperature at the exit of the furnace. For a good adhesion, the thermal treatment of the steel strips before their immersion in the liquid zinc is critical. A robust neural network was used to predict the speed of the steel strip in the furnace, thus providing a better control of the steel temperature (Pernía-Espinoza, Castejón-Limas, González-Marcos & Lobato-Rubio, 2005). Third, a mixed approach combining neural networks and genetic algorithms was used to predict both the furnace best working conditions along with the best strip speeds (Martínez-de-Pisón, Alba, Castejón & González, 2006). Fourth, we show the results of using a decision tree in order to determine which variables have a greater influence in the quality of the coating. This approach provided a set of rules that further helped in reducing the number of coils with defective zinc adhesion (Martínez-de-Pisón, Ordieres, Pernía, Alba, & Torre, 2007). Fifth, we show the results of applying an artificial lock at the skin-pass section to solve an infrequent problem that has serious consequences though: the incorrect labelling of the coils. The skin-pass is a section of the hot-dip galvanizing line that provides the metal with the desired mechanical properties and superficial finishing. This artificial lock (González-Marcos, Ordieres-Meré, Pernía-Espinoza, & Torre-Suárez, 2008) saves money by preventing the system from applying incorrect treatments to otherwise defective coils.

MAIN FOCUS

Data mining is, essentially, a process lead by a problem: the answer to a question or the solution to a problem is brought to light by data analysis. Different efforts have been made in order to formalize the practice of data mining. These split up the whole process into a number of sequential stages (Fayyad, Piatetsky-Shapiro, & Smyth, 1996; Pyle, 1999; Chapman, Clinton, Kerber, Khabaza, Reinartz, Shearer, & Wirth, 2000). Even though the name and definition of the stages are somewhat different in each model, the general concept is shared by all those proposals: first, the practitioner meets the data and becomes familiar with the problem at hand; then, the data is prepared and a number of models is built and evaluated; finally, the knowledge extracted is consolidated and ready to aid in the solution of the problem stated.

In real practice, the data mining process turns out to be highly dynamic and iterative, as new questions or ideas whose answer requires to be searched or implemented in former stages might occur at any time. The limit to the number of iterations is usually put by the nature of the case study, the availability of data sources, the knowledge of the needed tools, and the requirements and resources of the company (Martínez-de-Pisón, 2003).

Estimation of the Mechanical Properties of Galvanized Steel Coils

At the moment, the mechanical properties of the product to be manufactured (yield strength, tensile strength and elongation) cannot be measured directly but through slow and destructive laboratory tests performed after the galvanizing process. In such situation, an open loop control system must be used,

as a classical control strategy is not possible. To make matters worse, the mechanical properties of the galvanized steel coils can be altered along the whole manufacturing process, from the steel creation (determined by the chemical composition of the metal) to the moment at which the finished product is obtained (either as coils or sheets). Ordieres-Meré, González-Marcos, González & Lobato-Rubio (2004) built a model, based on neural networks, capable of predicting online the mechanical properties using the information provided by the manufacturing process sensors. The implementation of this model boosted the existing control systems and, eventually, lead to a significant improvement of the quality of the final product.

The approach applied to obtain the reported model followed a more general methodology: first, the data set was inspected and prepared using common visualisation techniques (histograms, scatter plots, etc.) and projection techniques (Sammon maps (Sammon, 1969) and principal components analysis, PCA (Dunteman, 1989)). Then, those samples marked as outliers were removed and the data was split up according to the identified clusters. Finally, the models for the mechanical properties were obtained.

To build the model a set of neural networks was trained. Neural networks are one of the most frequently used tools in industry owing to its efficiency and simplicity. Amongst the many types of neural networks currently known, one of the most adequate to build models that describe industry processes using their data as inputs and outputs is the multilayer perceptron (MLP). The reason why MLP are so much commonly used is because they can be trained to be universal approximators (Funahashi, 1989; Hornik, Stinchcombe. & White, 1989); a MLP with one hidden layer can be trained to approximate any continuous multivariate function, as long as the hidden layer contains as many neurons as needed to obtain the required precision.

The set of neural network had seventeen input neurons, a variable number of neurons in the hidden layer (between 3 and 20), and one output neuron. So as to prevent the overfitting effects (Haykin, 1998), the backpropagation with weight decay algorithm was chosen (learning parameter between 0.1 and 0.2; decay term between $5 \cdot 10^{-5}$ and $5 \cdot 10^{-7}$), since the number of train patterns was small for the dimension of the input layer. In order to obtain optimum generalization ability, early stopping was used. The database was divided into three sets of patterns: training (63.4% of the samples); validation (31.6% of available data); and test (the remaining 5%). The training patterns were used to adjust neuron weights, while the validation patterns were employed to check, at specific training cycles (thirty in this case from a total of one hundred thousand), that the error obtained with the training patterns was coherent with the error in the neural network compared with new data that would not have participated in the fitting process. When this validation error was minimised, network training was deemed to have concluded. Finally, the test patterns, which had never been seen by the trained network, were used to check its generalization ability. Moreover, to ensure that training results were not dependent on a specific selection, which might not have been ideal in terms of the content of its information due to its random nature, three sets of files were generated for training, validation and testing purposes. Finally, the network that best fitted the information contained in the data was selected.

To summarise, 324 experiments were generated for each variable to be predicted. After training all the networks, their behaviour in the test showed us the best configuration for each case studied. Out of the set of trained networks, the best configurations offered fairly good results, with relative average errors for the test patterns --- that is to say, those patterns not used during the training stage --- below 4.5% (see Table 1), a result within the tolerance accepted by the company. Even though better results can be expected by using a larger number of hidden neurons, the good results obtained demonstrate the feasibility to estimate online the properties of the coils being manufacture.

Table 1. Best networks configuration and test samples results for each mechanical property and cluster analysed.

Mechanical property	Parameter	Cluster1	Cluster2
Yield strength (MPa)	Learning rate	0.2	0.2
	Decay term	$5 \cdot 10^{-7}$	$5 \cdot 10^{-7}$
	Number of hidden units	17	7
	Mean relative error	3.19%	2.60%
Tensile strength (MPa)	Learning rate	0.2	0.2
	Decay term	$5 \cdot 10^{-7}$	$5 \cdot 10^{-7}$
	Number of hidden units	12	3
	Mean relative error	0.88%	1.82%
Elongation (%)	Learning rate	0.2	0.2
	Decay term	$5 \cdot 10^{-7}$	$5 \cdot 10^{-7}$
	Number of hidden units	18	7
	Mean relative error	3.07%	4.30%

The advantages of these models are self-evident, moreover when the time and money savings are considered: decisions can be made online and cheaply using better criteria. These decisions are mainly based on the predictions of the coil properties at every point: as the process variables are sampled along the whole length of the coil, the model obtained can be used to online draw a graph of the coil properties without any destructive laboratory test. Thus, the manufacturer can identify online those portions of the coil that do not meet the client's requirements and cut those portions beyond tolerance instead of having to discard the whole coil, with the huge savings derived from this strategy.

Annealing Furnace Band Speed Model

As it has already been said, one of the key processes to confer the adequate properties to the steel band and to get a good adherence of the zinc coating is the thermal treatment of the band before its dip in hot liquid zinc. The existing control system governing the temperature of the band in the annealing furnace only considered the reference temperature of the furnace. Martínez-de-Pisón (2003) proposed a more efficient control of the temperature. In order to obtain the best control of the temperature of the band at the exit of the heating area, the speed of the band needs to be controlled as well. Following this approach, and as a further enhancement to the annealing cycle control, Pernía-Espinoza, Castejón-Limas, González-Marcos & Lobato-Rubio (2005) proposed a robust model of the speed of the band at the annealing furnace that obtained more accurate temperatures at the exit of the furnace.

That speed model was built using data mining techniques. First, during the preparation of the data sets, it was noticed that the data set available contained a small fraction of samples, 3% approximately, with speed values beyond the normal range of operation. These values were due to transients in the line, as those that may occur when a coil is welded or when an atypical coil (with unusual dimension) is incorporated to the line. In such situations, when a decrease of the speed band is expected to happen, it seemed fit that the neural network should learn the relationships between the temperature and speed. Nevertheless, in order to avoid negative effects derived from considering those samples, robust models

seemed adequate to be used. Eventually, the robust network proposed by Liano (1996) was considered owing to its remarkable properties (high breakdown point and high efficiency on normally distributed errors) and the ease of implementation into the general backpropagation framework.

A set of MLP was considered again for training, in this case using seven input neurons, one output neuron and a variable number of neurons in the hidden layer (between 5 and 20). For training, the Fletcher-Reeves conjugated gradient optimization scheme was chosen. The best configuration had fifteen neurons in the hidden layer with a 4.43% relative average error on the test patterns (20% of the samples: 6184 patterns), which reflected the good behaviour of the model obtained and the feasibility of controlling the temperature of the coils at the exit of the annealing furnace in a more efficient manner. Moreover, this model can be used as well to establish the line operation strategy, to set the order in which the coils should be processed, predict the speed of the line in transient conditions, etc.

Neural Networks and Genetic Algorithms to Optimize the Annealing Process

The work reported by Martínez-de-Pisón, Alba, Castejón & González (2006) is another step in the enhancement of the control of the thermal treatment of the coils in the annealing furnace. The goal pursued was to obtain the best running parameters on the basis of the particular features of a given coil, the inertia of the annealing furnace and the transition areas between two coils with different features. For such purpose, neural networks and genetic algorithms were used.

First, the neural models were developed to define the process parameters in steady state. The patterns used for training those models contained information on the two types of control applied to the line: manual and automatic. Following this approach, the models learnt both from the experience of the machinist commanding the line and from the mathematical model integrated in the process. In this case, a set of MLP with three inputs (thickness, width and temperature of the band at the entrance of the furnace), a hidden layer with a variable number of neurons (between 2 and 25) and four outputs (temperatures at the entrance, centre and exit of the furnace, and the speed of the band), were trained. The learning algorithm used was backpropagation, with a learning parameter between 0.1 and 0.25. The training method used was 10-fold cross-validation: 80% of the samples were used for the training step and the other 20% (test patterns) were used to check the generalization ability of the neural network. Out of the 92 experiments generated, the best results obtained were fairly good. The error on the test patterns was 2.74%, both low and within tolerance.

Then, a model of the dynamic behaviour of the steel band under fluctuating temperature and speed conditions was built. As before, MLP lend a helping a hand. In this case, fifteen input neurons were needed: temperatures at the entrance, centre and exit of the annealing furnace, speed of the band, temperature of the band at the entrance and exit of the furnace, coil dimensions and a few others derived from these. As the variables of the data set were highly dependent, the PCA technique was used to reduce the dimension of the data set and hence the number of input neurons. With the new set of independent variables, nine in all, the model obtained provided errors close to 5.75% on the test patterns for the best network. As for the temperatures, no errors above 10°C were observed (see Figure 1.a).

Once the generalization capability of the former neural networks was demonstrated, the parameters of the furnace and the dynamic behaviour of the steel band were simulated. Then, by using genetic algorithms, the best operation conditions were searched trying to minimise the errors in the transition areas (changes in the dimensions, steel degrees, etc.) between coils. For such purpose, two new variables were considered: start and width of the transition. The algorithm was based on the following scheme:

1. Create 1000 chromosomes with random values for variables (first population) of position and length of the straight line adjustment of the temperature settings of zone 1, 2, 3 and 5 in the furnace and of the strip.
2. Using the settings of each chromosome, simulate the behaviour of the strip and obtain the maximum error instantaneously between the obtained temperature of the band and the actual temperature.
3. Obtain the top 20% best chromosomes (with the least error).
4. Create 70% new chromosomes with the crossover of the 20% best.
5. Create another 10% with mutations.
6. Create a new generation of 1000 chromosomes from those obtained in steps 3, 4 and 5.
7. Repeat steps 2 and 6, saving the results from each generation.

By creating seven generations, the results showed transitions in which the differences between the real and reference parameters decreased significantly. For example, the maximum temperature difference decreased from 50°C to 19.38°C (see Figure 1), and the average error from 15.1°C to 7.31°C. It is worth to mention that the methodology followed in this work can easily be ported to any industrial process as long as sufficient data is available.

Zinc Coating Adherence Problems Reduction

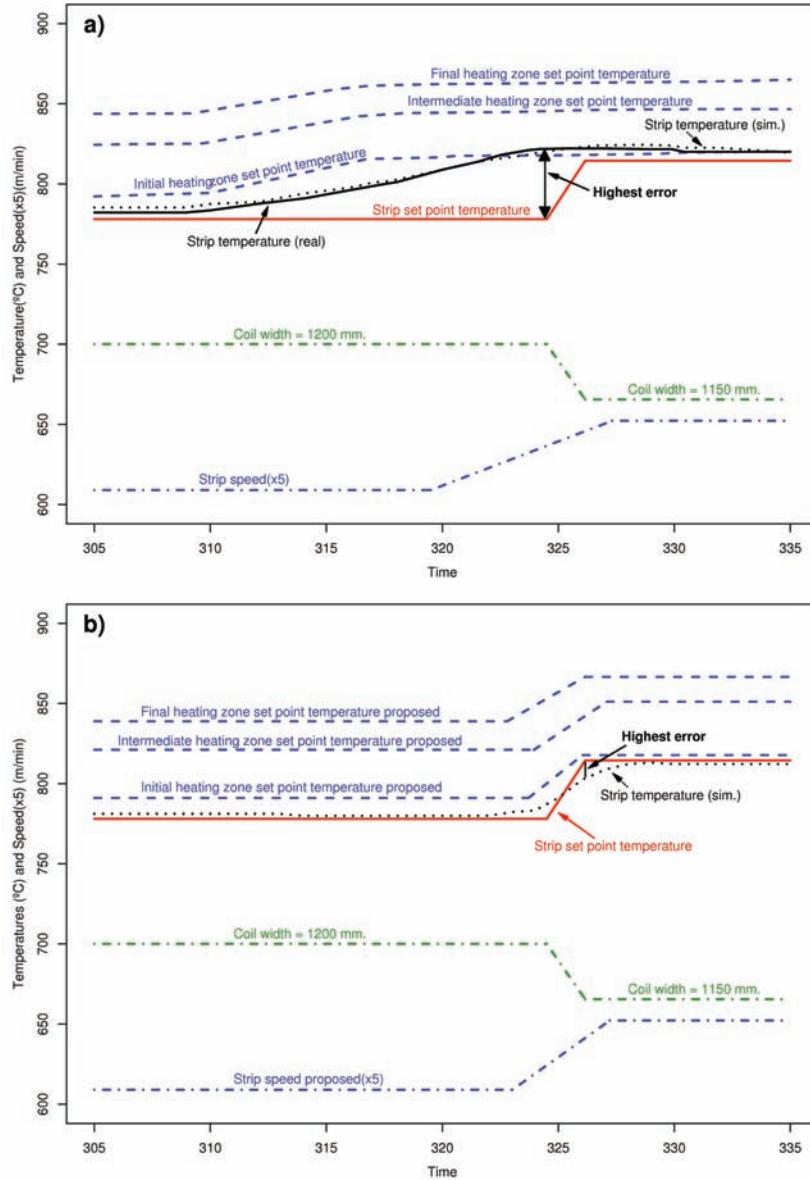
The experiences reported above focus on systems' modelling which, undoubtedly, constitutes one of the most interesting data mining projects in the industrial field. Nevertheless, data mining has much more to offer than just the modelling of systems. In the lines that follow, the work developed by Martínez-de-Pisón, Ordieres, Pernía, Alba & Torre (2007) is reported, where the identification of the causes that originated a quality deterioration of the zinc coating in new steels manufactured in the galvanising line under study --- hidden knowledge discovering --- by means of decision trees.

It is a well known fact that the quality, thickness and uniformity of the zinc coating are affected by multiple factors that we already have enumerated. The adjustment of these parameters for every type of coil is a hard task and, most frequently, the implicit knowledge generated at this stage is not documented as, being so many the number of variables to modify and adjust, it is very difficult to determine which actually have critical impact on the quality of the final product. It is here where the analysis of the historic logs of the process might help in extracting this knowledge and, thus, establish new control strategies that allow reducing the number of coils with irregular adherence.

In this case, the use of decision trees was justified by the large number of variables to analyse (more than 75) and the small number of existing defective coils (approximately 5%). Decision trees allow the use of data sets with a large number of variables and small number of samples. The search of hidden knowledge was performed through an iterative process of creation and analysis of decision trees. For every model, an analysis was run to evaluate the information of those branches that provided a better separation of the coils with adherence problems.

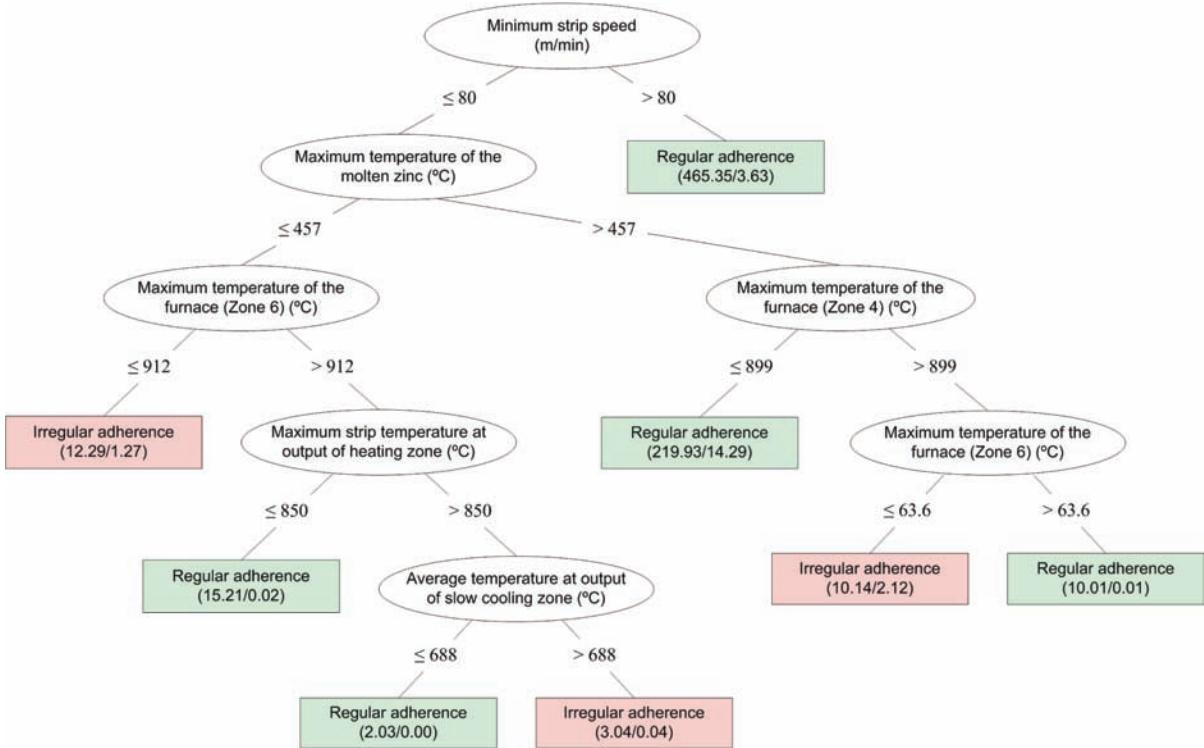
The results obtained were visualised and contrasted with the opinions of the experts of the plant to determine the physical or logical causes of the defects, as many times the conclusions derived from the trees were obvious or useless. Along the work, some of the conclusions accorded with the previous knowledge of the experts: the changes in the speed of the strip and the temperature of the molten zinc were the variables with strongest impact in the adherence. Nevertheless, the methodology used provided an added value since besides of confirming the influence of these variables in the adherence problem,

Figure 1. Example of optimization by means of neural networks and genetic algorithms. a) Measured and predicted strip temperature with initial parameters; b) Predicted strip temperature (dotted line) with proposed parameters.



it defined the values for these parameters that discriminated the coils with irregular adherence. In addition, several relationships amongst variables were discovered as, for example, the speed of the strip, the temperature of the molten zinc and the temperature of the furnace (Figure 2), that could be very interesting to provide some light in the understanding of process internals.

Figure 2. First decision tree generated.



Skin-Pass Artificial Lock

As an example of the spawning of new perceptions, ideas, etc., that the data mining process generates, we report here the development of an artificial lock at the skin-pass section. The skin-pass is a section of the line that bestows the material with the adequate mechanical and surface finishing. The idea of developing this artificial lock was born as a consequence of the good results obtained in the prediction of the mechanical properties of the coils. The goal pursued, in this case, was to find a solution to a problem that occurs rarely, but whose consequences are grave: the incorrect labelling of the steel degree of a coil. If a client receives a coil with a wrong label, their machinery can suffer severe damage, the end user can receive a product with lower resistance which can result in accidents, etc.

In order to detect these errors, an artificial lock for the skin-pass was developed (González-Marcos, Ordieres-Meré, Pernía-Espinoza, & Torre-Suárez, 2008). This lock consisted in a model to predict online the elongation of the steel coils at the skin-pass on the basis of the data coming from the sensor of the process and the chemical composition of the steel. Thus, a significant difference between the estimated and the real elongation would send an alert signal to separate that coil and proceed with further detailed analyses to determine its condition.

Once more, the data mining general methodology was applied. After the analysis and initial preparation of the data sets, different neural networks (MLP) were trained using the backpropagation algorithm to obtain a selection of the best models.

For each of the four steel classes analysed, a set of perceptron neural networks with twenty six inputs, a hidden layer with a variable number of neurons (between 20 and 48) and one output, were trained. The

Table 2. Best networks configuration and results for each steel class analysed.

Steel class	Total samples	Hidden units	Test mean absolute error
0	1022	30	0.0874
1	6282	31	0.0712
2	2003	22	0.0652
4	1484	24	0.1285

hidden neurons had a logistical activation function, whereas the output function was linear. The learning algorithm used was backpropagation, with a learning parameter equal to 0.2. A regularisation method was also used in training to avoid overfitting problems in the trained network: weight decay, with a decay term equal to $5 \cdot 10^{-7}$. Again, early stopping was used in order to obtain optimum generalization ability and the database was divided into three sets of patterns: training (57% of the samples); validation (28% of available data); and test (the remaining 15%).

The small errors obtained in the prediction of the elongation at the skin-pass (Table 2) demonstrated the quality of the models but not the quality of the lock. That is to say, in order to use the models obtained as wrong labelling indicators it was mandatory that the models were able, not only of providing good estimates of the elongation for the good coils, but of providing significantly different estimates when the chemical composition was incorrect. So as to check at the production line the usefulness of the models obtained to solve the problem at hand, their behaviour was tested with coils known to be incorrectly labelled. In these cases, the observed and estimated elongation were significantly different, with errors ranging from -0.334 to -1.224. Following this approach, it was possible to test the feasibility of the lock and to establish as well the threshold to send an alert signal.

The advantages of this approximation are self-evident, since the integration of this lock in the productive process prevents the manufacturing of wrongly labelled coils and its consequences. In addition, we must mention the ease of implementation of the models built, as the software used for training the neural networks (SNNS, Stuttgart Neural Network Simulator) can easily export the neural networks to C programming language code.

FUTURE TRENDS

Data mining opens a world of new opportunities for many industries. The iron and steel making sector has already taken advantage of a few of the tools the data mining has to offer, successfully applying them in an effective manner at otherwise very hard problems. But the box has just been opened, and many more advanced tools and techniques are on their way to come. Fortunately, the industry has already recognized the benefits of data mining and is eager to exploit its advantages.

CONCLUSION

Data mining is useful wherever data can be collected. Of course, in some instances, cost/benefit calculations might show that the time and effort of the analysis is not worth the likely return. Obviously,

the idea of ‘Data Mining wherever and whenever’ is not advisable. If an application requires a model that can be provided by classical tools, then that is preferable insofar as this procedure is “less energy consuming” than those linked to DM methodologies.

We have presented various types of problems where data management can yield process improvements and where usually they are not so frequent due to the non-direct way of implementation.

In spite of these difficulties, these tools can provide excellent results based on a strict methodology like CRISP-DM (Chapman, Clinton, Kerber, Khabaza, Reinartz, Shearer, & Wirth, 2000) in combination with an open mind.

Data mining can yield significant, positive changes in several ways. First, it may give the talented manager a small advantage each year, on each project, with each customer/facility. Compounded over a period of time, these small advantages turn into a large competitive edge.

REFERENCES

- Abonyi, J., & Feil, B. (2007). *Cluster analysis for data mining and system identification*. Berlin, Germany: Birkhäuser Basel.
- Bellman, R. (1961). *Adaptive control processes: A guided tour*. Princeton, NJ: Princeton University Press.
- Bloch, G., Sirou, F., Eustache, V., & Fatrez, P. (1997). Neural intelligent control for a steel plant. *IEEE Transactions on Neural Networks*, 8(4), 910–918. doi:10.1109/72.595889
- Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *CRISP-DM 1.0: Step-by-step data mining guide*. Retrieved from <http://www.crisp-dm.org>
- Díaz Blanco, I., Cuadrado Vega, A. A., Diez González, A., Rodríguez Loredo, L., Obeso Carrera, F., & Rodríguez, J. A. (2003). Visual predictive maintenance tool based on SOM projection techniques . *Revue de Métallurgie-Cahiers D Informations Techniques*, 100(3), 307-315.
- Domínguez, O., Miranda-Ariz, J. M., & Salvador, L. (2002). Efecto protector de las capas de productos de corrosión de exposición atmosférica. *Revista de Metalurgia*, 38(2), 108–116.
- Dunteman, G. H. (1989). *Principal components analysis*. Newbury Park, CA: Sage Publications.
- Fabrellas, B., Ruiz, M. L., Martínez, M., Sanz, P., & Larrazábal, D. (2002). *Evaluación de la generación de dioxinas y furanas en el sector de la galvanización en caliente en el año 2002*.
- Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). Knowledge discovery and data mining: Towards a unifying framework. In E. Simoudis, J. Han, & U. Fayyad (Eds.), *Proceeding of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)* (pp. 82-88). Menlo Park, CA: AAAI Press.
- Funahashi, K. (1989). On the approximate realization of continuous mapping by neural networks. *Neural Networks*, 2, 183–192. doi:10.1016/0893-6080(89)90003-8
- Goldberg, D., & Sastry, K. (2007). *Genetic algorithms: The design of innovation*. Berlin, Germany: Springer.

- González-Marcos, A., Ordieres-Meré, J. B., Pernía-Espinoza, A. V., & Torre-Suárez, V. (2008). Desarrollo de un cerrojo artificial para el skin-pass en una línea de acero galvanizado por inmersión en caliente. *Revista de Metalurgia*, 44(1), 29–38. doi:10.3989/revmetalm.2008.v44.i1.93
- Haykin, S. (1998). *Neural networks: A comprehensive foundation*. Upper Saddle River, NJ: Prentice Hall.
- Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359–366. doi:10.1016/0893-6080(89)90020-8
- Institute for parallel and distributed high performance systems. (n.d.). *SNNS – Stuttgart neural network simulator. User manual, ver. 4.2* [online]. Retrieved from <http://www-ra.informatik.uni-tuebingen.de/downloads/SNNS/SNNSv4.2.Manual.pdf>
- Liano, K. (1996). Robust error measure for supervised neural network learning with outliers. *IEEE Transactions on Neural Networks*, 7(1), 246–250. doi:10.1109/72.478411
- Martínez-de-Pisón, F. J. (2003). *Optimización mediante técnicas de minería de datos del ciclo de recocido de una línea de galvanizado*. Unpublished doctoral dissertation, University of La Rioja, Spain.
- Martínez-de-Pisón, F. J., Alba, F., Castejón, M., & González, J. A. (2006). Improvement and optimisation of a hot dip galvanizing line using neural networks and genetic algorithms. *Ironmaking & Steelmaking*, 33(4), 344–352. doi:10.1179/174328106X101565
- Martínez-de-Pisón, F. J., Ordieres, J., Pernía, A., Alba, F., & Torre, V. (2007). Reducción de problemas de adherencia en procesos de galvanizado mediante técnicas de minería de datos. *Revista de Metalurgia*, 43(5), 325–336. doi:10.3989/revmetalm.2007.v43.i5.77
- Ordieres-Meré, J. B., González-Marcos, A., González, J. A., & Lobato-Rubio, V. (2004). Estimation of mechanical properties of steel strips in hot dip galvanizing lines. *Ironmaking & Steelmaking*, 31(1), 43–50. doi:10.1179/030192304225012060
- Pernía-Espinoza, A. V., Castejón-Limas, M., González-Marcos, A., & Lobato-Rubio, V. (2005). Steel annealing furnace robust neural network model. *Ironmaking & Steelmaking*, 32(5), 418–426. doi:10.1179/174328105X28829
- Pyle, D. (1999). *Data preparation for data mining*. San Francisco: Morgan Kaufmann Publishers.
- Rendueles, J. L., González, J. A., Díaz, I., Diez, A., Seijo, F., & Cuadrado, A. (2006). Implementation of a virtual sensor on a hot dip galvanizing line for zinc coating thickness estimation. *Revue de Métallurgie-Cahiers D Informations Techniques*, 103(5), 226-232.
- Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, 18(5), 401–409. doi:10.1109/T-C.1969.222678
- Schiefer, C., Jörgl, H. P., Rubenzucker, F. X., & Aberl, H. (1999). Adaptive control of a galvannealing process with a radial basis function network. In *Proc. of the 14th IFAC World Congress, Vol. 0*, Beijing, PR China (pp. 61-66).

- Tenner, J., Linkens, D. A., Morris, P. F., & Bailey, T. J. (2001). Prediction of mechanical properties in steel heat treatment process using neural networks. *Ironmaking & Steelmaking*, 28(1), 15–22. doi:10.1179/030192301677803
- Tian, Y. C., Hou, C. H., & Gao, F. (2000). Mathematical modelling of a continuous galvanizing annealing furnace. *Developments in Chemical Engineering & Mineral Processing*, 8(3/4), 359–374.
- Wang, P. P., Ruan, D., & Kerre, E. E. (2007). *Fuzzy logic: A spectrum of theoretical & practical issues*. Berlin, Germany: Springer.

KEY TERMS AND THEIR DEFINITIONS

Artificial Neural Network: A set of connected artificial neurons.

Artificial Neuron: A mathematical model that generally applies a non-linear function to a linear combination of the input variables in order to provide an output value.

Closed Loop Control System: A system that utilizes feedback as a mean to act over the driving signal according to the differences found amongst the observed and expected behavior.

Decision Tree: A classifier in the form of a tree structure, where each node is either a leaf node or a decision node. A leaf node indicates the value of the target attribute (class) of examples.

Genetic Algorithms: A systematic method used to solve search and optimization problems and apply to such problems the principles of biological evolution, namely, selection of the ‘fittest’, sexual reproduction (crossover) and mutation.

Multilayer Perceptron: A specific kind of artificial neural network whose neurons are organized in sequential layers and where the connections amongst neurons are established only amongst the neurons of two subsequent layers.

Learning Algorithm: In the artificial neural network area, the procedure for adjusting the network parameters in order to mimic the expected behavior.

Open Loop Control System: A system that does not rely on feedback to establish the control strategies.

Outlier: An observation that does not follow the model or pattern of the majority of the data.

Overfitting: Fitting a model to best match the available data while loosing the capability of describing the general behaviour.

Robust: Not affected by the presence of outliers.

Steady-State Regime: Status of a stabilized system.

Transient Regime: Status of a system that is changing from one steady-state regime to another.

Chapter 21

Application of Neural Networks in Animal Science

Emilio Soria

Valencia University, Spain

Carlos Fernández

Valencia Polytechnic University, Spain

Antonio J. Serrano

Valencia University, Spain

Marcelino Martínez

Valencia University, Spain

José R. Magdalena

Valencia University, Spain

Juan F. Guerrero

Valencia University, Spain

ABSTRACT

Stock breeding has been one of the most important sources of food and labour throughout human history. Every advance in this field has always led to important and beneficial impacts on human society. These innovations have mainly taken place in machines or genetics, but data analysis has been somewhat ignored. Most of the published works in data analysis use linear models, and there are few works in the literature that use non-linear methods for data processing in stock breeding where these methods have proven to obtain better results and performance than linear, classical methods. This chapter demonstrates the use of non-linear methods by presenting two practical applications: milk yield production in goats, and analysis of farming production systems.

DOI: 10.4018/978-1-60566-766-9.ch021

INTRODUCTION

The use of neural networks (NN) has undergone an exponential increase during the last few years due to the different types of problems that can be solved; they can be successfully applied to classification, modelling and prediction problems. Their inherent characteristics when compared to other methods make them especially suitable in data analysis problems where some of the following conditions occur:

1. It is unclear or very hard to find out the rules that relate the target variable to the other variables considered in the model.
2. Data are incomplete, imprecise or noisy (statistically perturbed).
3. The problem requires a great number of dependant variables (problems with high dimensionality).
4. The model to be applied is non linear.
5. There exists a great amount of data.
6. The environment of the variable or variables to model changes with time.

The above-mentioned characteristics are quite common and representative in a great number of knowledge areas, but in accordance with the experience of the authors, animal science fits these characteristics particularly well for several reasons:

1. Any animal and its possible interactions with the different elements in its environment constitute a complex system with a large number of plausible relationships. In principle, it would be natural to assume these relationships as non linear, due to the inherent complexity of living beings.
2. There are many variable to determine the state/behaviour of an animal or to model one of the variables that characterizes the animal. Therefore, an excessive simplification of the problem may yield too many errors in the model. Consequently, this is a non-linear, high dimensionality problem.
3. Data gathering is performed, in most of the cases, by the farmer himself, so incomplete data or errors in measures can arise.
4. Since this kind of data comes from a daily event (i.e., feed amount, dairy amount, ...), growth over time, a system that can adapt to new data gathered, and can also bring reliability and performance to the new yielded results would be a desirable choice.

The above information suggests not only the possibility of applying neural networks in this field but also their suitability for problems of this kind. In this chapter, we will present two applications based on these models. The first one is a problem of time-series prediction, where the target is to predict goat milk production; the second one is a data clustering application: the farm milk and meat productions are analyzed and characterized. The results obtained, which are compared with other classic approaches, demonstrate the validity of the neuronal models in animal science.

WEEKLY MILK PREDICTION ON DAIRY GOATS

The aim of this problem is to predict the milk production in a goat herd. It should be taken into account that farmers' income comes from milk production and composition; therefore, the accurate measurement

or prediction of milk yield (MY) is essential to their economy (Pool & Meuwissen, 1999; Macciotta, Cappio-Borlino & Pulina, 2000). Also, it is interesting to determine and evaluate the most important factors that can affect milk yield, so that it is feasible to interact with all the variables that decisively improve milk production.

Neural networks have been applied in a great number of time-series prediction problems demonstrating the validity of these methods over other approaches (Weigend & Gerschenfeld, 1994). According to the aforementioned characteristics, two stages can be defined in this problem:

1. To develop a neural model (namely a multilayer perceptron) that yields accuracy in solving the problem and, that (unlike most approaches in animal science), can be applied to any animal. Instead of working in a sequential way a parallel and non-linear approach is proposed, in order to gain flexibility.
2. Once the best models are obtained, a sensitivity analysis of the inputs will be carried out in order to determine the variables that most contribute to the milk yield prediction; these will be the factors to be taken into account in order to optimize the milk yield for a given herd.

A homogeneous and representative group of dairy goats on a commercial farm, which is a member of ACRIMUR (Murciano-Granadina Goat Breeder Association), was selected to implement the predicting neural models. Although a great number of variables affect milk performance, just a few variables are available under practical conditions. Experts in animal science and veterinary specialist have selected the following ones as essential input variables (Gipson & Grossman, 1990):

- **Number of lactations:** indicates the age of the goat in our case. As significant difference in milk yield is found between primiparous and adults.
- **Litter Size:** indicates the number of kids per goat. Goats have high prolificacy, and it is easy to obtain 2-3 kids per goat and birth. Besides, milk yield is greater when a high number of births are obtained.
- **Number of controls:** just indicates the number of milk yield records obtained during the lactation period. In our study, we registered one milk yield record per week and goat.
- **Metabolic Weight:** is the body weight of the animal in kilograms raised to the power of 0.75.
- **Milk yield:** is the milk production of the goat expressed in kilograms per day.
- **Days between parturition and the first milk yield control:** goats have a gestation period of 5 months, but the flock does not have the partum at the same time for all goats, so the number of days between the partum and the first control is an important variable to take into account since it will affect the total milk yield.
- **Diet:** is an important factor to consider. The level of energy, protein and nutrients will influence the amount and quality of the milk obtained.

Of the 35 goats, 22 goats were used to build the neural model (all goats were fed with the same diet but different sources of protein; 11 goats were fed with soybean meal, and the other 11 were fed with sunflower meal) and the remaining 13 goats (6 goats fed with soybean meal and 7 goats fed with sunflower meal) were used to validate the model. It is important to emphasize that these 13 goats were not used to build the model in order to demonstrate the generalization capability of the network. This is a usual choice in the NN literature, i.e., to use two thirds of the patterns to obtain the model, and the other

Table 1. Mean and standard deviation (s) for the training and validation data set

	<i>Mean</i>	σ
Training group (n=22)		
Number of lactations	4.87	1.12
Litter size	1.96	0.63
Number of controls	8.68	2.71
Metabolic weight, kg	17.20	1.49
Milk yield, kg/d	2.21	0.82
Validation group (n=13)		
Number of lactations	4.96	1.64
Litter size	1.81	0.49
Number of controls	7.92	3.92
Metabolic weight, kg	16.77	1.14
Milk yield, kg/d	2.10	0.65

third of the patterns to validate it. The objective of modelling the lactation curve is generally to predict the milk yield each day of lactation with minimum error. Several empirical models have been developed to describe such a lactation curve for milk production, and all of them lack the flexibility to adapt to different management or environmental conditions. This fact implies that the models proposed in the literature have few generalization capabilities, which is one of the key characteristics that applications of this kind must exhibit. Table 1 shows the average value and standard deviation for input variables and desired output (milk production).

Table 1 reflects that the division of the initial set into two subsets is valid because mean values and standard deviations are quite similar in both subsets. Once the input variables were chosen and the data divided, variable values were standardized (zero mean and variance equal to one); this normalization keeps the MLP from giving primacy to some inputs for their range instead of for their importance in solving the problem (Bishop, 1995).

Figure 1. Structure of a neuron; the w_k are known as synaptic weights, and they are the parameters of the network; x_i are the inputs to the neuron; w_0 is called bias or threshold.

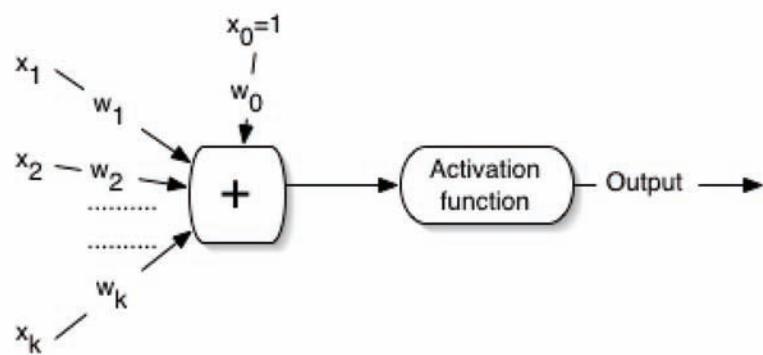
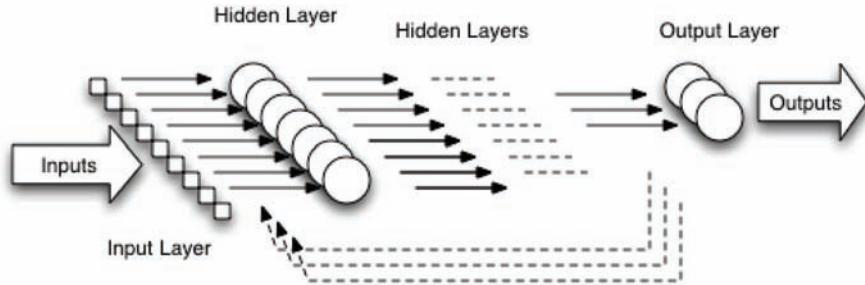


Figure 2. Structure of a multilayer perceptron. Every circle in the figure represents a neuron as described in Figure 1.



Neural Model Applied

The NN used in this work is the Multi-Layer Perceptron (MLP). This model consists of a layered arrangement of individual computation units known as artificial neurons. The neurons of a given layer feed the neurons of the next layer with their outputs. A single neuron is shown in Figure 1 (Haykin, 1998)

The inputs x_k to a neuron are multiplied by adaptive coefficients w_k , that are called synaptic weights, which represent the connectivity between neurons. The output of a neuron is usually taken to be a sigmoid shape (sigmoid or hyperbolic tangent function). Thus, it yields:

$$o_j = \phi\left(\sum_{i=0}^m w_{ij} \cdot x_i\right) \quad (1)$$

Neurons from a specific network are grouped together in layers that form a fully connected network. The first layer contains the input nodes, which are usually fully connected to hidden neurons (neurons located between the first and last layers) and these are, in turn, connected to the output layer. In this case, only one output neuron is necessary, since only one variable is predicted at each time. NN can be trained using on-line or batch adaptation (Haykin, 1998; Bishop, 1995). Figure 2 shows a diagram of a fully connected multilayer perceptron.

The first choice in the development and training of the model is the selection of the learning algorithm. This stage offers plenty of options (Durenberger, 1984) where, in most cases, a local search algorithm (usually gradient-based methods) is used due to its simplicity in implementation and greater speed than, for example, global search algorithms (genetic algorithms). In this work, the momentum-backpropagation algorithm (Bishop, 1996) was chosen, over other more complex algorithms in the same family because of its simplicity and similar efficiency. Care must be taken with this method since it finds the closest minimum; thus, it becomes necessary to develop a set of neural models with different initial values of the weights in order to obtain different adjustments.

The next choice is the learning model. There are two main options in this stage: *on-line* and *batch learning*. On-line learning consists of modifying the weights before figuring out the error committed by the neural network for each example; *batch learning* updates the weights when all the data is presented to the model before knowing all the errors committed by the network for every pattern in the input subset

(Haykin, 1998). On-line learning is the best one to model variables that present abrupt variations, while batch learning determines models where the outputs are smoother, without abrupt variations. Note that, in many applications, those variations come from the measurement instruments and are not due to the problem itself. For this reason, it is advisable to use both learning types since the use of batch mode decreases the contributions of rare cases to the model, whereas the on-line mode can consider contributions of noise or error in data. Both approaches are considered in this work.

The previous choices are related to the learning algorithm; the next choice is the MLP structure. The number of hidden layers and the number of neurons in each layer also affect the modelling and generalization capacity of the NN. It is possible to demonstrate mathematically that two hidden layers are sufficient to solve any problem (Ripley, 1996; Haykin, 1998; Orr & Müller, 1998). In this work, NN with one and two hidden layers have been considered. Once NN models with one hidden layer were analysed, we trained models with two hidden layers. In this case, the results (not shown) were poorer than the ones obtained using a single hidden layer. It must be emphasized that increasing the number of free parameters (namely the number of neurons or layers of the network) does not mean that a better result is going to be achieved, since the phenomena of overfitting can appear (Haykin, 1998; Bishop 1995). In order to avoid this problem it is common to stop the training by cross validation (early stopping). In other words, the data set is divided into two subsets. One of the subsets is used for training the neural network, and the second one is used to check the performance with data that is different from the training data with two hidden layers, the capability of network modelling is increased but early stopping also becomes more difficult (Haykin, 1998).

Analysis of the Importance of the Inputs

As we have mentioned above, apart from making an accurate prediction of the yielded-milk values, (which will help to determine the evolution of farm incomes), the goal of this work is to determine the most influential variables in this prediction. This way we will have the variables that can be tuned in order to optimize the performance of the farm.

One of the most important problems that neural networks have been blamed for in the literature is being a “black-box”; that is, we can establish accurate models, but we cannot infer any knowledge regarding the problem from them. This fact has proven to be erroneous (Pal, 1999; Zurada & Cloete, 2005); one of the possible ways of extracting knowledge is to carry out what is known as *sensitivity analysis* (Orr & Muller, 1998). The idea is quite simple and can be applied to any model. Let us suppose in principle that you want to determine the importance of a given input (lets say the j -th input). First the output for every available pattern is calculated; the output for pattern k is designed as O_k . Then, the value of the variables whose importance is going to be calculated is zeroed out for every pattern in the input; then the output of the network for this modified pattern is calculated, and each of these outputs will be denoted as S_k . The last stage is to evaluate the expression:

$$I_j = \sum_{k=1}^N |O_k - S_k| \quad (2)$$

The more this value tends to zero, the less important the input is. The idea behind this method is very simple: values close to zero in equation (2) mean that it does not matter whether or not the variable is

taken into account. This indicates that the variable does not contribute to the calculation of the output of the neural network in a significant way.

Obtained Results

The objective of this problem is the prediction of a time series (we must predict the future milk-yield), so if a MLP is used, there are two alternatives (Weigend & Gerschenfeld, 1994):

- *To use a model that is specialized in time-series.* In this approach, MLPs are used in which, the architecture of the network describes the nature of the problem to be solved. This approach yields recurrent models (Elman, Jordan networks) or networks where the synaptic weights become digital filters (IIR or FIR filters); (Wan, 1993)
- *To take into account the dynamics of the inputs.* In this approach, delayed values of the desired signal are also considered as inputs to the system; it is the non-linear equivalent to an autoregressive model, a classical model in time-series prediction (Ljung, 1999; Makridakis, Wheelwright, & Hyndman, 1997).

The second approach is the one applied in this work for several reasons. The most relevant reason is that due to the structure of the network, the sensitivity analysis cannot be performed. The second reason is the simplicity of development and implementation of the neural model. We have developed two kinds of NN models; one that uses the MY of the current week and one that uses current and previous MY. Therefore, four models were developed according to the type of learning and its temporal characteristics. We must also point out that every one of the neural initial configurations is trained a certain number of times (100 times in this particular problem) with different synaptic weight initialization, to avoid the problem of local minima. With this method, a great number of neural models were obtained (8400 different models) from which those that offered the best performance in milk-yield prediction were selected.

In order to study the goodness of the NN models, four performance indexes were taken into account: Mean Square Error (MSE), Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Error (ME). The first three indexes measure the accuracy of the obtained fitting, and the last one, ME, points out the possible bias in the neural model of the prediction. The slope of the straight line between the desired signal and the network output (labelled with a) is also provided; in a perfect model this value should be close to one. Figures 3 and 4 show these indexes for training and validation sets, respectively.

Using these figures show that a good fitting is obtained when the standard deviation in milk production as measure for every goat in both subsets (training and validation). The only goat with problems in the fitting is number 13 in the validation subset (with the greatest error as well as a negative index in the fitting of the desired signal to network output).

In order to compare the accuracy of the three models, Table 2 shows the four indexes: for the best neural model, (whose given value is the average of all the data set); for the simplest prediction model which considers the predicted value as the current value; and for an auto regressive model, which considers only one delay when applying Akaike criteria (Ljung, 1999).

Table 2 shows that the best results in the validation set (which defines the generalization capabilities of the model) are obtained with the developed neural model. This proves the suitability of the proposed

Figure 3. Error indexes for the training set (22 goats) obtained with the neuronal model with the least generalization error.

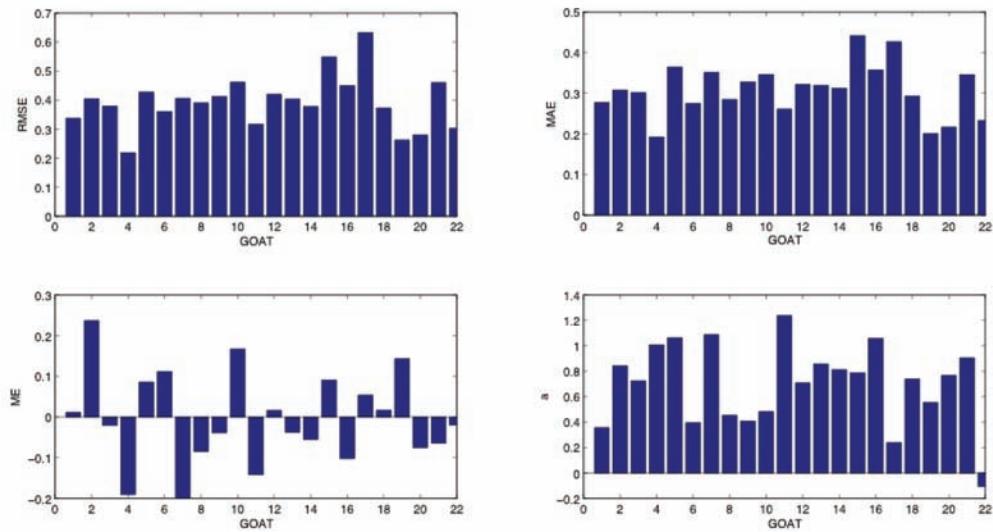


Figure 4. Error indexes for the generalization set (13 goats) obtained with the model that yields the least error.

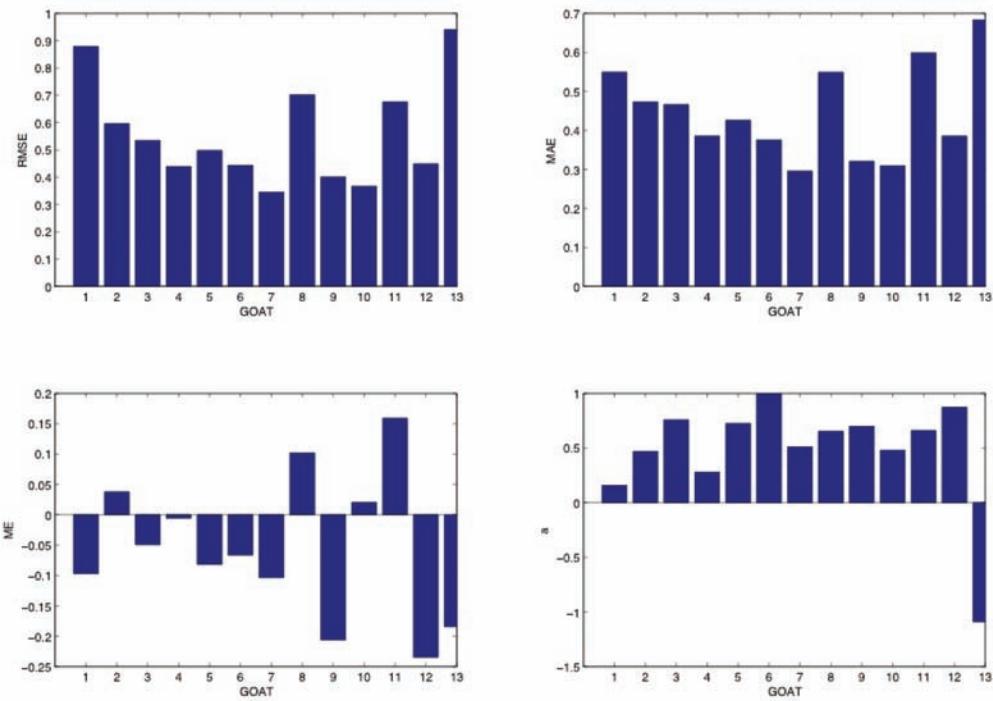


Table 2. Performance indexes for the best neural model obtained, a trivial model [MY(t+1)=MY(t)] and a first-order autoregressive model [AR(1)].

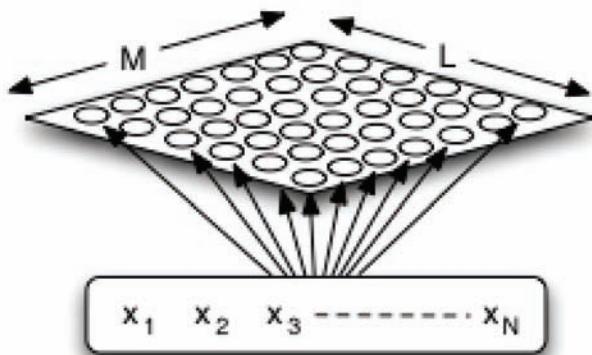
	MAE	RMSE	ME	a
Best NN				
Validation Set	0.28	0.36	-0.03	0.83
Training Set	0.25	0.32	0.00	0.73
MY(t+1)=MY(t)				
Validation Set	0.32	0.42	-0.02	0.79
Training Set	0.29	0.38	0.03	0.90
AR(1)				
Validation Set	0.30	0.41	-0.02	0.70
Training Set	0.28	0.37	-0.02	0.80

solution. The next step is to establish the ranking of importance of the different inputs considered in the neural model for milk-yield prediction. In order to obtain this ranking, the 10 best neural models (those with a smaller RMSE in the validation stage) were taken into account and expression (2) was calculated for every input. The final step is to order the variables according to the calculated value (descending), which is in the order of importance of the variables in this problem. These results show that the MY prediction depends mainly on the current and the previous values of MY, which indicates the importance of these values; it is a logical result, since these inputs implicitly describe the dynamics of the process. The next most important input for the training data set is the days between parturition and the first control, and the third most important is the type of diet. The least important inputs were metabolic weight, litter size and number of lactations because as we explained above we selected a small numbers of goats that belonged to the same farm, and had homogeneous number of lactations, litter size and body weight.

ANALYSIS OF LIVESTOCK MANAGEMENT USING THE SOM

The importance of goats as providers of essential food in meat and dairy products around the Word has been documented (Haenlein, 2004). More than any other mammalian farm animal, the goat is a main supplier of dairy and meat products for rural people (home consumption). The second aspect of demand for goat milk is the connoisseur's interest in goat milk products, especially cheeses and yoghurt in many developed countries. This demand is growing because of the increasing levels of disposable income. Therefore, any business or enterprise activity has a series of related variables whose knowledge and relation is essential for good management. This knowledge of relations and variables is also necessary for all cattle operations where live organisms have high variability to changes in environmental conditions. Besides economic factors, biological factors must also be taken into account, which can make it more difficult to address optimal management of cattle operations (Daza *et al.*, 2004). The current objective now is to extract all the possible knowledge of the data that can be compiled for goat operations in order to improve them; thus, we are trying to solve a classic problem of Data Mining, which is specifically known as data clustering (Hand, Mannila & Smyth, 2001). These techniques consist of determining groups within our data. These clusters can allow us to determine profiles that, at first glance, we would

Figure 5. SOM structure



not be able to determine. This segmentation of data is very useful in problems where trends within a data set are expected. The neural models that we will use are known as Self Organizing Maps, SOM, (Kohonen, 2001; Haykin, 1998) which we describe briefly below.

Self Organizing Maps

The model is based on the grouping of similar tasks that takes place in the different zones of the brain (Kohonen, 2001). In this model, the neurons are organised in a two-layers architecture. The first layer is the input layer that consists of m neurones, (one for each input variable). The processing is performed in the second layer or competition layer, (Figure 5):

A weight vector that has the same dimension as the input vector is assigned to every neuron in the second layer. The final goal is to get similar patterns in inputs to increase the activity of nearby neurons in the output layer of the SOM. One possible way is to establish a similarity function between the input vector and the weight vectors that are associated to each output neuron. This is where a problem arises: what do we consider as similar? To make the system work, we must define valid similarity markers. There are different ways of measuring these similarities, Table 2 shows the ones that are most widely used.

A weight vector is assigned to every neuron in the output layer. This vector is also called a synaptic weight. These coefficients characterize every neuron. The underlying idea in this neural model is that *neighbour patterns in the N-dimension space should also be neighbours in the space determined by the output layer in the SOM* (Figure 5 shows a bi-dimensional space). Thus the SOM is also applied to carry out the task of extracting knowledge from the variable. The SOM also preserves topological relationships among data while mapping data into a lower dimensional map. By means of different visual representations, in the output space of lower dimensionality, we can infer relations among the variables in the input space. There are several different learning algorithms for this neural network, but one of the most frequently used is described below (Kohonen, 2001; Haykin, 1998; Arbib, 2002):

1. Initialization of synaptic weights, which can be performed in several ways: assigning the values of an input as initial weight values or, simply, initializing them randomly. For the neuron ij (located in the i -th row and the j -th column in the competition layer) we can define its weight vector as

$$w_{ij} = [w_{ij}^1, w_{ij}^2, \dots, w_{ij}^N]$$

2. Input of a input pattern in every iteration; the R-th pattern is denoted as $x_R = [x_R^1, x_R^2, \dots, x_R^N]$.
3. Calculation of the similarity among weights of each neuron and the input pattern. If the Euclidean distance is considered as the comparison measure, the calculation is performed for all the neurons in SOM as $d(w_{ij}, x_R) = \sum_{k=1}^M (w_{ij}^k - x_R^k)^2$
4. Determination of the Winner Neuron (WN). It will be the one with the smallest value of $d(w_{ij}, x_R)$.
5. Update of the synaptic weights; if the chosen similarity function is the quadratic function, then $w_{ij} = w_{ij} + \alpha \cdot h(w_{ij}) \cdot (x_R - w_{ij})$: where α is a parameter related to the learning rate and $h(w_{ij})$ is the neighbourhood function. The output value of the neighbourhood function depends on the distance between the Winner Neuron and the neuron corresponding to the updating coefficients (with an amplitude peak in the winner neuron).
6. If the maximum number of iterations is reached, end; otherwise go to step 2.

The key to this algorithm lies in steps 3-6; in these steps, we can determine which neuron assigns the input pattern in the current iteration (step 3); obviously, we assign the most similar weight vector. With the update in step 5, we perform two tasks: we modify the weight vector so that it resembles the input vector and afterwards, we place the neighbourhood of the winner neuron closer to this input pattern (since the neighbourhood function decreases with distance). Thus, similar zones in the input space correspond to closer zones in the bidimensional space in the output layer in SOM.

SOM extracts information from different elements: (Kohonen, 2001).

- *Values of the synaptic weights.* The learning algorithm is, in short, a classic clustering algorithm with small modifications to take advantage of the information that the output layer in SOM can provide. The representatives or centroids in the clustering correspond to the values of the synaptic weights.
- *Distribution of neurons in the output layer.* This kind of network maps the input patterns to different zones in the output layer, so the zones that corresponds to the input patterns can be visualized, thus inferring relations among input patterns/variables.

Data Used in the Study

The data used in this study refer to 1999 and were obtained between 2001 and 2002 from surveys returned by 90 farmers dedicated to goat farming as their main activity. The surveyed farms comprise approximately 15% of the total of the official census of Murciano-Granadina breed farms in the Murcia Region. The farm samples were obtained using a stratified sampling by counties: Altiplano (14 surveys), Campos de Cartagena (19 surveys), Noroeste (19 surveys), Río Mula (3 surveys), Valle del Guadalentín (30 surveys), Vega Del Segura (5 surveys). The questionnaire comprised different sections; land bases (Murcia Region, unfavourable area), herd composition (herd size; sheep and/or goats), management practice (groups, artificial nursing, prolificacy), milking machine, milking type), feeding management (use of agricultural bioproducts, compound feed, total mixed ratio (TMR), automatic feeding, milking feed supplementation) and production type (milk and meat).

Following the advice of agronomists, nutritionists and veterinarian experts, we decided to use the following 10 input variables, which are mainly related with feeding practices. This is one of the greatest costs for farmers due to the fact that the raw material for animal feeding is being used for biofuels. Therefore, there is less food available for animal feeding and it has higher prices.

- **Study Area:** is where the farm is geographically located; in our study, Murcia Region (Spain).
- **Goat Herd Size:** is the number of goats per farm ranging between fewer than 50 goats per farm and more than 400 goats per farm.
- **Animal Feeding:** indicates if the farmer prepares his own diet for the animal or buys compound feed.
- **Body Condition Score (BCS):** this variable is related to the body reserves of the animal. If the farmer controls the BCS of his goats, it indicates better management and possibility of supplementation when BCS is low.
- **Milking type:** this variable evaluates how the farmer does the milking process. Some farmers do not have milking machines others do manual milking. Most dairy farmers have milking machine.
- **Refrigeration Tank:** this variable indicates if the farmer has a refrigeration tank on the farm. This is indirectly associated with milking machines, which means that a farmer with a refrigeration tank also has milking machines.
- **Replacement Feeding:** indicate if farmer take into account the feeding of the future breeders.
- **Milk Feeding:** indicates whether or not the farmer pays special attention to the goats in lactation because more milk production requires more nutrients in the diet.
- **Gestation Feeding:** this indicates whether or not the farmer takes into account the feeding during gestation, because the foetus growth during the last part of the gestation period is exponential and requires extra nutrients.
- **Other Species:** indicate if the farmer has just dairy goats or any other species.

It is worth noting that the data were standardized (mean zero and variance unity) in order to avoid unbiased models due to the different range of values of the variables. We should point out certain aspects about the architecture and learning of the SOM. First, for the architecture a map was developed where neurons were hexagons. The reason is because, with this geometric shape, the number of neighbours to every neuron increases, so a better representation of data on the map was obtained. Second, for the number of neurons, a bidimensional map made up by 50 neurons (10×5) was developed. As mentioned above each neuron was represented by a vector, which consisted of a number of components, which were established fixed by the number of questions in our survey (10 dimensions). Figure 5 shows the values of the different components for the synaptic weight using a grey scale. Each hexagon represents a neuron and the dark shades indicate low values, negative or absence of a variable characteristic; white shades indicate greater values and positive or presence of a variable characteristic. This representation helps recognize zones where the variables take high or low values and, therefore, helps in extracting qualitative rules about the data.

It must be noted that, in addition to the fact that the weight vector of the neurons represents the centroids of the clustering of the data, we can visualize the actual situation of all the components studied and the relationships and interactions among them.

Feeding becomes an important factor to take into account when a new stockbreeding activity is initi-

ated since it typically involves more than half of the farming expenses (Haenlein, 2001). The components *Milking Type*, *Replacement Feeding* and *Milk Feeding* are shown in the upper area of the maps (Figure 6) with white shades (more than half of the rectangle) indicating that farms allocated in those areas have milking machines and supplement the diet of both, replacement and milking. The shades of the variable *Refrigeration Tank* is in upper rectangle area as well, which corresponds to farms that have milking machines and manage feeding, as we mentioned above. *Refrigeration Tank* has 2 possible options, and this is the only variable that has different codification; farms with a refrigeration tank show dark shades and code 0 (which coincidences with the light areas in the variables *Milking Type*, *Replacement Feeding* and *Milk Feeding*), with the meat herds having white shades and code 1 (this is the reason why it seems that farms with milking machines do not have a refrigeration tank, (which is not true, it is just a different codification). That means that the white area for *Refrigeration Tank* (lower right) corresponds to meat production animals. If we pay attention to the variable *Other Species*, is possible to observe that the white area indicates other species, meaning that, in the lower-right part of the map, farmers include sheep for meat purposes.

We found that the main type of activity was dairy milk production and that most farmers have the Murciano-Granadina breed with milking machines and a medium size herd (between 100 and 200 goats). These type of farm is similar to those found in Southeastern Spain (Andalusia) which has the Malagueña goat breed, where infrastructures and installations for milking machines are, in general, good (Castel *et al.* 2003). In Andalusia, Mena *et al.* (2005), Ruíz *et al.* (2008) found higher intensification in dairy milk production when the herd is not free ranging (the farmer that has a concentrated feeding system has a more intensive level of production). In the Valencian Community with Murciano-Granadina breeds, Fernández *et al.* (2008a) also found a tendency to intensification, which is linked to feed support by specialists, with 70% of the goat's production more than 300 kg of milk per goat and lactation. These types of farms are similar to those found in the Murcia Region, where the situation of dairy farms has improved considerably (installations and management) since the description of 1995 by Falagán *et al.* (1995). The last study in the Murcia Region was developed by Navarro (2005), which showed a greater tendency to intensification and specialization in milk production. The intensive production system usually has good nutrition management (mainly using TMR and compound feed). SOM has also shown farms that are specialized in meat production with sheep (in our study, the Segureña breed) and large herd size that is specialized in lamb production. Milan *et al.* (2003) also found the same tendency to intensification when the breed was used for meat production (Ripollésa breed).

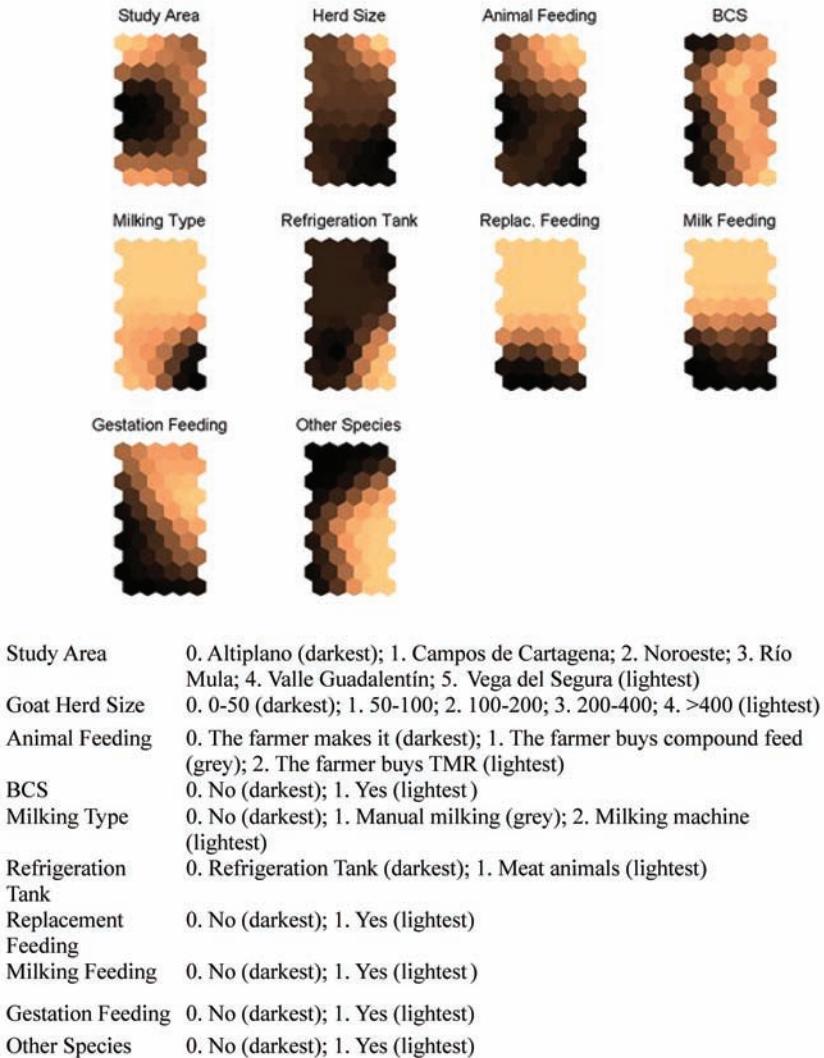
It can be concluded that most farmers who supplement the animals' diets (compound feed or TMR) tend to be those with milking machines and refrigeration tank facilities and a herd size of around 200 goats. On the other hand, farms that do hand milking had lower herd sizes with no nutritional management.

In summary, SOM represents an important visualizing tool for a better understanding of the reality of the productive system, allowing us to optimize the processes involved.

CONCLUSION

In this chapter, we have presented the application of neural networks, which is one of the most important trends in machine learning, applied to animal science. In this area, there are a great number of problems: time-series prediction problems (Fernández *et al.*, 2002; Fernández *et al.*, 2004); pattern classification

Figure 6. Representation of the components of the synaptic weight vectors corresponding to each neuron in the map. A grey scale code is used to represent the values of these components.



(Navarro and Fernández, 2006); system modelization (Fernández *et al.*, 2008b); as well as pattern clustering. Most of the previous works in the literature use linear models (multivariate analysis and logistic regression) as well as classical clustering methods (*k*-neighbours). These approaches are not optimal since data are related to very complex systems that are hard to model, i.e., animals and possible interactions with the associated variables. Therefore, a new approach to the resolution of problems in this area using neuronal networks is presented, in order to affirm their suitability over other methods, we have considered their use in two different problems: prediction of time series (obtaining the future milk production of a goat) and clustering (analysis of farming management operations by means of the data collected with surveys) (Sanzogni and Kerr, 2001; Grzesiak *et al.*, 2006, Fernández *et al.*, 2006;

Fernández *et al.*, 2007a, b; Fernández *et al.*, 2008a). The obtained results have demonstrated the effectiveness of these methods in this area of knowledge and, we encourage interested readers to use other machine learning methods.

REFERENCES

- Arbib, M. (2002). *The handbook of brain theory and neural networks*. Cambridge, MA: MIT Press.
- Bishop, C. M. (1995). *Neural networks for pattern recognition*. Gloucestershire, UK: Clarendon Press.
- Castel, J. M., Mena, Y., Delgado-Pertiñez, M., Carmuñez, J., Basalto, J., & Caravaca, F. (2003). Characterization of semi-extensive goat production systems in southern. *Small Ruminant Research*, 47, 133–143. doi:10.1016/S0921-4488(02)00250-X
- Falagán, A., Guerrero, J. E., & Serrano, A. (1995). Systèmes d'elevage caprin dans le sud de l'Espagne. In: *Goat production systems in the Mediterranean* (pp. 38-50). Wageningen, The Netherlands: Wageningen Pers.
- Fernández, C., Gómez, J., Sánchez Seiquer, P., Gómez-Chova, L., Soria, E., Moce, L., & Garcés, C. (2004). Prediction of weekly goat milk yield using autoregressive models. *South African Journal of Animal Science*, 34, 165–168.
- Fernández, C., Martínez, B., Gómez, E., Peris, C., Pascual, J. J., Serrano, A. J., & Soria, E. (2008a). Quantitative analysis of official milk control in Valencia community by SOM. In *Proceedings of the 9th International Conferences on Goats* (p. 106).
- Fernández, C., Pascual, J. J., Blas, E., & Cervera, C. (2008b). Milk prediction in dairy goats using multicomponent system model. *J. Appl. Anim. Res.*.
- Fernández, C., Sánchez, A., & Garcés, C. (2002). Modeling the lactation curve for test-day milk yield in Murciano-Granadina goats. *Small Ruminant Research*, 46, 29–41. doi:10.1016/S0921-4488(02)00179-7
- Fernández, C., Soria, E., Mardalena, R., Martín, J. D., & Mata, C. (2007b). Qualitative analysis of feed management practice on goats herds by SOM in Murcia Region. *Journal of Applied Animal Research*, 32, 41–48.
- Fernández, C., Soria, E., Martín, J. D., & Serrano, A. J. (2006). Neural network for animal science applications; two case studies. *Expert Systems with Applications*, 31, 444–450. doi:10.1016/j.eswa.2005.09.086
- Fernández, C., Soria, E., Sánchez-Seiquer, P., Gómez-Chova, L., Magdalena, R., & Martín-Guerrero, J. D. (2007a). Weekly milk productions on dairy goats using neural networks. *Neural Computing and Application Journal*, 16, 373–381. doi:10.1007/s00521-006-0061-y
- Gipson, T. A., & Grossman, M. (1990). Lactation curves in dairy goats: A review. *Small Ruminant Research*, 3, 383. doi:10.1016/0921-4488(90)90019-3

- Grzesiak, W., Blaszczyk, P., & Lacroix, R. (2006). Methods of predicting milk yield in dairy cow; predictive capabilities of Wood's lactation curve and artificial neural network. *Computers and Electronics in Agriculture*, 54, 69–83. doi:10.1016/j.compag.2006.08.004
- Hand, D. J., Mannila, H., & Smyth, P. (2001). *Principles of data mining*. Cambridge, MA: MIT Press.
- Haykin, S. (1999). *Neural networks: A comprehensive foundation*. Upper Saddle River, NJ: Prentice Hall.
- Ljung, L. (1999). *System identification. Theory for the user*. Upper Saddle River, NJ: Prentice Hall.
- Luenberger, D. (1984). *Linear and nonlinear programming* (2nd ed.). Reading, MA: Addison Wesley.
- Macciotta, N. P. P., Cappio-Borlino, A., & Pulina, G. (2000). Time series autoregressive integrated moving average modelling of test-day milk of dairy ewes. *Journal of Dairy Science*, 83, 1094–1103.
- Makridakis, S. G., Wheelwright, S. C., & Hyndman, R. J. (1997). *Forecasting: Methods and applications*. New York: John Wiley & Sons.
- Mena, Y., Castel, J. M., Caravaca, F. P., Guzmán, J. L., & González, P. (2005). Situación actual, evolución y diagnóstico de los sistemas semiestensivos de producción caprina en Andalucía centro-occidental. In *Junta de Andalucía. Consejería de agricultura y pesca* (pp. 222).
- Milán, M. J., Arnalte, A., & Caja, G. (2003). Economic profitability and typology of Rapollesa breed sheep faros in Spain. *Small Ruminant Research*, 49, 97–105. doi:10.1016/S0921-4488(03)00058-0
- Navarro, M. J. (2005). *Caracterización socio-económica de los sistemas de producción de caprino de la comunidad autónoma de Murcia*. Unpublished doctoral dissertation, Univ Miguel Hernández, Elche, Spain.
- Navarro, M. J., & Fernández, C. (2006). Introduction to the situation of the goat sector in Murcia Region. In *Options Mediterranees. Serie A 70* (pp. 157-163).
- Orr, G. B., & Müller, K. R. (1998). *Neural networks: Tricks of the trade*. Berlin, Germany: Springer-Verlag.
- Pal, N. (1999). Soft computing for feature analysis. *Fuzzy Sets and Systems*, 103, 201–202. doi:10.1016/S0165-0114(98)00222-X
- Pool, M. H., & Meuwissen, T. H. E. (1999). Prediction of daily milk yield from a limited number of test days using test day models. *Journal of Dairy Science*, 6, 1555–1564.
- Ripley, B. D. (1996). *Pattern recognition and neural networks*. Cambridge, UK: Cambridge University Press.
- Ruiz, F. A., Castel, J. M., Mena, Y., Camuñez, J., & González-Redondo, P. (2008). Application of the technico-economic analysis for characterizing, making diagnoses and improving pastoral dairy goatsystems in Andalusia (Spain). *Small Ruminant Research*, 77, 208–220. doi:10.1016/j.smallrumres.2008.03.007
- Sanzogni, L., & Kerr, D. (2001). Milk production estimates using feed forward artificial neural networks. *Computers and Electronics in Agriculture*, 32, 21–30. doi:10.1016/S0168-1699(01)00151-X

Wan, E. (1993). Time series prediction using a neural network with embedded tapped delay-lines. In A. Weigend & N. Gershenfeld (Eds.), *Predicting the future and understanding the past, SFI studies in the science of complexity*. Reading, MA: Addison-Wesley.

Weigend, A. S., & Gerschenfeld, N. A. (1994). *Time series prediction: Forecasting the future and understanding the past*. Reading, MA: Addison-Wesley.

Zurada, J. M., & Cloete, I. (2005). *Knowledge-based neurocomputing*. Cambridge, MA: MIT Press.

KEY TERMS AND DEFINITIONS

Dairy Goat: Goat specialized in milk production.

Dairy Farmer: Farmer that raises milk producing animals.

Multilayer Perceptron: The most well known and most widely used neural model in problems such as: systems modelization, time-series prediction and, pattern classification. The name stands for the location of the neurons in several layers.

Self Organizing Maps (SOM): Neural model that is mainly used, to extract knowledge from data with high dimensionality. These maps can be visualized in two dimensions showing relations among data.

Recurrent Neural Networks: Neural models where the synaptic weights contain feedback. This network exhibits dynamic temporal behaviour.

Milk Yield: Milk production expressed in Kg per animal and day.

Chapter 22

Statistical Machine Learning Approaches for Sports Video Mining Using Hidden Markov Models

Guoliang Fan

Oklahoma State University, USA

Yi Ding

Oklahoma State University, USA

ABSTRACT

This chapter summarizes the authors' recent research on the hidden Markov model (HMM)-based machine learning approaches to sports video mining. They will advocate the concept of semantic space that provides explicit semantic modeling at three levels, high-level semantics, mid-level semantic structures, and low-level visual features. Sports video mining is formulated as two related statistical inference problems. One is from low-level features to mid-level semantic structures, and the other is from mid-level semantic structures to high-level semantics. The authors assume that a sport video is composed of a series of consecutive play shots each of which contains variable-length frames and can be labelled with certain mid-level semantic structures. In this chapter, the authors present several HMM-based approaches to the first inference problem where the hidden states are directly associated with mid-level semantic structures at the shot level and observations are the visual features extracted from frames in a shot. Specifically, they will address three technical issues about HMMs: (1) how to enhance the observation model to handle variable-length frame-wise observations; (2) how to capture the interaction between multiple semantic structures in order to improve the overall mining performance; (3) how to optimize the model structure and to learn model parameters simultaneously. This work is the first step toward the authors' long-term goal that is to develop a general sports video mining framework with explicit semantic modeling and direct semantic computing.

DOI: 10.4018/978-1-60566-766-9.ch022

INTRODUCTION

Video mining is to discover knowledge, structures, patterns and events of interest in the video data, and its benefits range from efficient browsing and summarization of video content to facilitating video access and retrieval in a large database or online multimedia repository. There are various types of video data. According to different production and edition styles, videos can be classified into two main categories: *scripted* and *non-scripted* videos (Xiong, Zhou, Tian, Rui, & Huang, 2006). Scripted videos are produced according to a certain script or plan that are later edited, compiled, and distributed to users for consumption. News and movies are the examples of scripted videos. Most research efforts on scripted videos focus on the development of a Table-of-Content (TOC) that provides an overview of the video's naturally organized content to support efficient browsing and indexing. On the other hand, the events in non-scripted videos happen spontaneously and usually in a relatively fixed setting, such as meeting videos, sports videos, and surveillance videos. The research on non-scripted videos involves detecting high-lights and events-of-interest. In this chapter sports video mining is studied using American football as the case study.

Although sports videos are non-scripted content, there are still definite or repetitive structures and patterns. Using these structures and patterns, we can develop some flexible and effective tools for video browsing/indexing. Currently, there are two kinds of approaches for sports video mining, *structure-based* (Kokaram, et al., 2006; Xie, Chang, Divakaran, & Sun, 2003) and *event-based* (Assfalg, Bertini, Colombo, Bimbo, & Nunziati, 2003; T. Wang, et al., 2006). The former uses either supervised or unsupervised learning to recognize some basic semantic structures (such as canonical view in a baseball game or play/break in a soccer game) that can serve as an intermediate representation supporting semantics-oriented video retrieval, but usually cannot deliver high-level semantics directly. The latter one provides a better understanding of the video content by detecting and extracting the events-of-interest or the highlights, which could be very specific and task-dependent and requires sufficient and diverse training examples. Because these two approaches are complementary in nature, researchers have investigated how to integrate both of them in one computational framework. For example, a mid-level representation framework was proposed for semantic sports video analysis involving both temporal structures and events hierarchy (L. Y. Duan, Xu, Chua, Q. Tian, & Xu, 2003) and a mosaic-based generic scene representation was developed from video shots and used to mine both events and structures (Mei, Ma, Zhou, Ma, & Zhang, 2005).

Machine learning is one of the most feasible approaches for semantic video analysis (Kokaram, et al., 2006). Hidden Markov models (HMMs) have been the most popular tool in this area (Xie, Chang, Divakaran, & Sun, 2003). Our goal is try to establish a HMM-based machine learning framework that supports both structure analysis and event analysis and delivers the rudimentary building blocks for high-level semantic analysis. There are two major distinctions in our research. One is the concept of semantic space which supports explicit semantic modeling and specifies both mid-level and high-level semantic structures as well as relevant visual features, so that the task is formulated as two inference problems. The other is a new HMM that offers multi-faceted advantages for semantic video analysis.

In our recent research (Ding & Fan, 2006, 2007a, 2007b, 2008), we have discussed several HMM-based approaches that are able to support unsupervised learning for semantic modeling and computing. We assume that an American football video is composed of a series of consecutive play shots with all breaks and commercials removed. We are interested in two mid-level semantic structures that include “*play types*” (what happened) and “*camera views*” (where it happened), both of which are defined at the shot-level and can be specified by a set of frame-wise visual features in a shot. Although these seman-

tics may not necessarily directly reflect highlights or events of interest, they are considered as the most common mid-level descriptors for most field sports and can provide an intermediate representation to explore high-level semantics when used together. For example, a shot in the central view of a long play followed by a shot in the end zone is likely to be a “touchdown” highlight. We have proposed several new extended HMMs that offer more flexibility, capability and functionality than existing HMMs for video mining. Specifically, we have discussed the following three main technical issues related to HMMs: (1) how to enhance the observation model in the HMM to handle variable-length frame-wise observations for shot-wise state inference; (2) how to capture the interaction between multiple semantic structures in order to improve the overall mining performance; (3) how to optimize the model structure and to learn model parameters simultaneously? Although our research is focused on American football videos, it could be applied to other sport video mining applications. The major advantage of our approach is the explicit semantic modeling and direct semantic computing that lead to a general mid-level representation for sport video mining.

This chapter is organized as follows. We will first review the recent work on HMM-based semantic video analysis where we discuss the development on HMM theory. Secondly, we introduce the concept of semantic space where relevant semantic structures are specified. Then we discuss the visual features that are used to infer semantic structures. We present several HMM-based approaches and compare them in terms of their modeling capability, efficiency and the video mining performance. Finally, we conclude this chapter by providing some discussion for future research.

RELATED WORK

The machine learning approaches to video mining are generally classified into two categories. One is the *discriminative approach* that involves a parametric model to represent the posterior probabilities of the hidden states given the observations. The other is the *generative approach* that estimates posterior probabilities of hidden states by involving conditional densities (likelihood) for observations given priors.

The discriminative approaches, e.g., the conditional random field (CRF) (Lafferty, McCallum, & Pereira, 2001) and the support vector machines (SVM) (Sadlier & Connor, 2005), tend to directly estimate the posterior probability of latent states given observations. They are expressive due to the fact that they support flexible statistical dependency structures. In general, the discriminative approaches are appreciated by their simplicity and efficiency in various video analysis problems. However, they may be sensitive to noise or limited when the data set is complex (Nallapati, 2004).

Generative approaches are usually preferred for a complex data set when some prior knowledge is available such as the distribution of observations. Compared with discriminative ones, they rely on a dynamic model on hidden states and an observation likelihood function, and the decision is made via Bayesian rules. The Dynamic Bayesian Network (DBN) (Murphy, 2002) provides a unified probabilistic framework to represent various graphical structures with directly dependency, and the Hidden Markov Models (HMM) (P. Chang, Han, & Gong, 2002) is considered as the simplest DBN that has been widely used in many video analysis applications. There have been intensive efforts to enhance HMMs for semantic video analysis, and the recent studies mainly focus on two issues, i.e., *structuring* and *learning*.

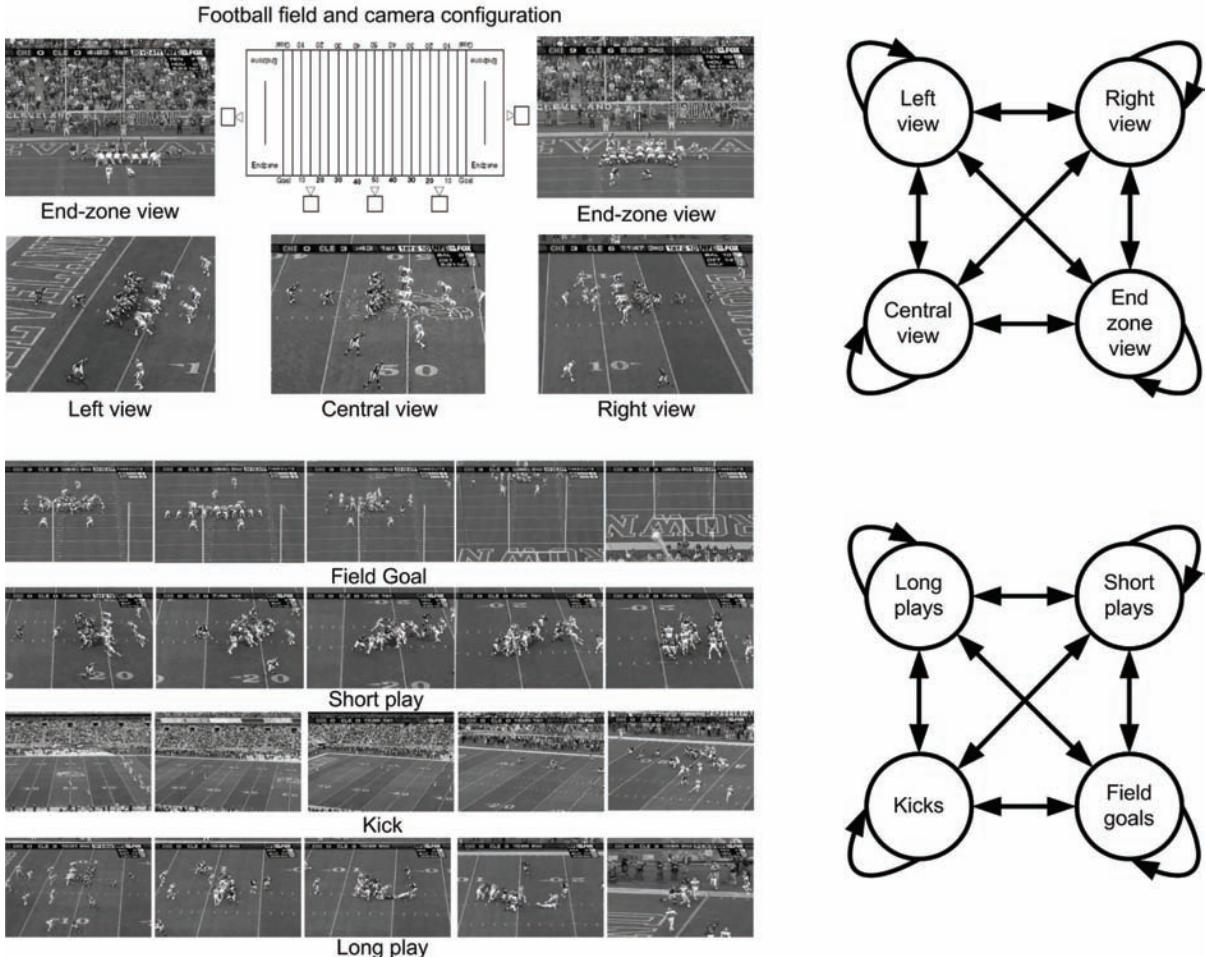
Two kinds of statistical structures have been added to the traditional HMM framework to enhance its capability and flexibility, the *parallel* and *hierarchical* structures. A parallel structure involves information fusion at either the decision-level or the feature-level. For example, when there are two parallel

hidden structures involved in a video sequence, the Coupled HMM (M. Brand, Oliver, & Pentland, 1997) and the Influence Model (Zhang, Gatica-Perez, Bengio, & Roy, 2005) were proposed to capture the interaction between two Markov processes via decision-level fusion. Differently, the Factorial HMM includes multiple uncoupled state transitions that share the same observation sequence (feature-level fusion) (P. Wang & Ji, 2005) or where observation vectors are obtained by concatenation of low-level features from different modalities or sources (Nefian, Liang, Pi, Liu, & Murphy, 2002). On the other hand, the hierarchical structure usually imposes a multi-layer representation where event detection can be accomplished by two steps, *recognition of primitives* and *recognition of structures*. For example, the Hierarchical HMM (HHMM) is able to capture low-level primitives based on which we can represent some mid-level structures, e.g., plays and breaks (Xie, et al., 2003). Furthermore, some signal processing applications may desire a more flexible observation model that can represent variable-length observations for state estimation. For example, the segmental HMM was proposed for speech recognition that can effectively handle variable-length observations by involving a segmental observation model (Gales & Young, 1993). The key idea in the SHMM is a two-layer observation model that captures feature variability both within a segment and across segments.

There are two learning issues about HMMs or DBNs. One is the learning of model structures (Friedman & Koller, 2003), and the other is the learning of model parameters (Ghahramani, 2002). The former one tries to provide a compact and effective model representation by condensing the state space and eliminating unnecessary state dynamics. The latter one aims at estimating the model parameters given certain model structure. Although these two issues are often addressed separately, they are not independent. Recent research reveals that it is imperative to have simultaneous structure learning and parameter learning for complex DBNs or HMMs to ensure their effectiveness and efficiency. For example, when the HHMMs were used for unsupervised sports video mining, the reverse-jump Markov Chain Monte Carlo (RJMCMC) method is embedded in the learning of HHMMs to support model selection and feature selection (Xie, et al., 2003). The concepts of entropic prior and parameter extinction were proposed to simplify and optimize the HMM structure by a state trimming process (Matthew Brand, 1999). The goal is to find a compact model structure for the HMM that has good determinism and minimal ambiguity, i.e., a sparse and concise state space.

In our research, we are interested in developing new HMMs that offer more capacity, functionality and flexibility than existing HMMs. The new HMMs should be able to take advantage of explicit semantic modeling where two mid-level semantic structures defined at the shot level directly correspond to the latent states and visual features extracted at the frame-level are used as the observations. We expect the new HMMs should be able to accommodate rich statistics of frame-wise observations for shot-wise state estimation. Also, we want to explore the mutual interaction between two semantic structures by integrating two Markov processes into the same inference process. Moreover, as the modeling complexity goes up, model training becomes more challenging and even problematic. We need a more powerful learning algorithm that can optimize the model structure and learn the model parameters at the same time. Although recently proposed HMMs can deal with these issues individually, we want to develop an integrated approach that is able to address all above issues in one framework.

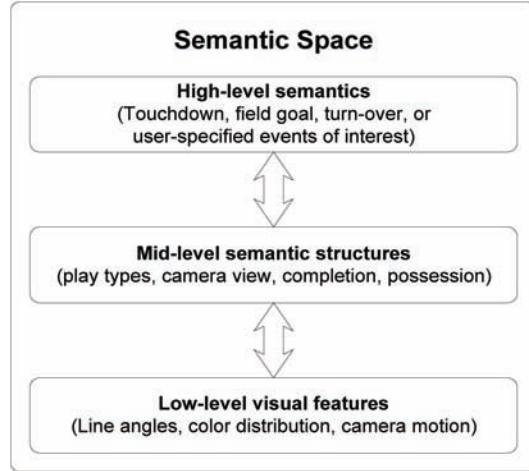
Figure 1. Two semantic structures and their dynamics in the American football video, i.e., camera views and play types.



PROBLEM FORMULATION

The major challenge for sports video mining is the *semantic gap* between low-level visual features and high-level semantic concepts. There are two methodologies to address this issue, i.e., *data-driven bottom-up methods* and *concept-driven top-down approaches*. A balanced interaction between both methods should be encouraged to provide a generic representation for semantic video analysis (Calic, Campbell, Dasiopoulou, & Kompatsiaris, 2005). Specifically, we argue that the top-down concept-guided semantic representation plays a critical role in sports video mining that can be supported by data-driven bottom-up methods in an inference framework. Therefore, we first define some basic mid-level descriptors in a sport video. For example, in an American football game (or other field-based games), there are two common mid-level semantic structures: “camera views” and “play types”, which construct rudimentary semantics in a game. In this work camera views include the *central*, *left*, *right*, *end-zone* views, and play types cover the *long*/*short plays*, *kick* and *field goal*, as shown in Figure 1, where we also show the dynamics within

Figure 2. Semantic space for American football videos.

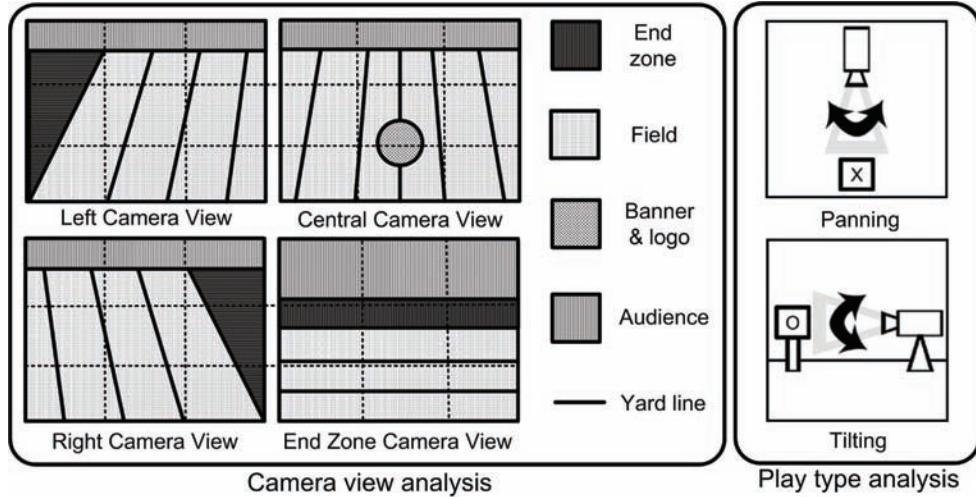


each semantic structure. There could be several other mid-level semantic structures, e.g., the possession (offense/defense) and the play completion (complete/incomplete). As our future research, these mid-level semantic structures can be treated as the building blocks for high-level semantic analysis.

We want to advocate a concept of semantic space that is formed by a set of rules and used to facilitate the high-level understanding of video content and user queries (Campbell, 2006; S. F. Chang & Hsu, 2006; Natsev, Naphade, & Smith, 2004). We specify a three-layer semantic space as shown in Figure 2, where we want to use *mid-level semantic structures* to represent *high-level semantics*, such as highlights, and mid-level semantic structures are specified by relevant *visual features* extracted from the video data. It is our belief that the exploration of mid-level semantic structures from low-level visual features is more reliable and feasible than directly inferring high-level semantics from low-level features. Our main reasoning is that mid-level semantic structures have a relatively stable and repeated pattern and high-level semantics should be inferred from mid-level semantic structures due to their inherent relationship. For example, in American football games, high-level semantics includes highlights, e.g., a touchdown (TD), extra point after TD, field goal, turnovers, etc, as well as any event-of-interest that could be customized by the user. Directly detecting them may be challenging due to their complex nature and diverse variability. On the other hand, some mid-level semantic structures can be useful to represent and detect high-level semantics. Similar video mining paradigms can be found in recent literature, (L.-Y. Duan, Xu, Chua, Tian, & Xu, 2003; T. Wang, et al., 2006). It is our attempt to provide a unified machine learning paradigm where semantic video analysis is formulated as a statistical inference problem, i.e., *we want to infer mid-level semantic structures from low-level features, so that we can further infer high-level semantics from mid-level semantic structures*.

Moreover, we expect to have three major advantages in this framework with explicit semantic modeling. First, we can fully take advantage of the available semantics and prior knowledge about the rules in a sport game that makes the video mining problem well structured and formulated. Second, it provides both structure analysis (at the mid-level semantics) and event analysis (at the high-level semantics) that are complementary in nature for semantic video analysis. Third, it not only supports highlight detection but also is able to deliver customized events-of-interest, increasing the usability and interactivity of video data. Particularly, our recent research is focused on exploring mid-level semantic structures, e.g.,

Figure 3. Visual features used for video mining: Right. Color distribution and yard line angle. Left: Camera motion.



“camera views” and “play types”, from low-level visual features, and we have achieved some promising results.

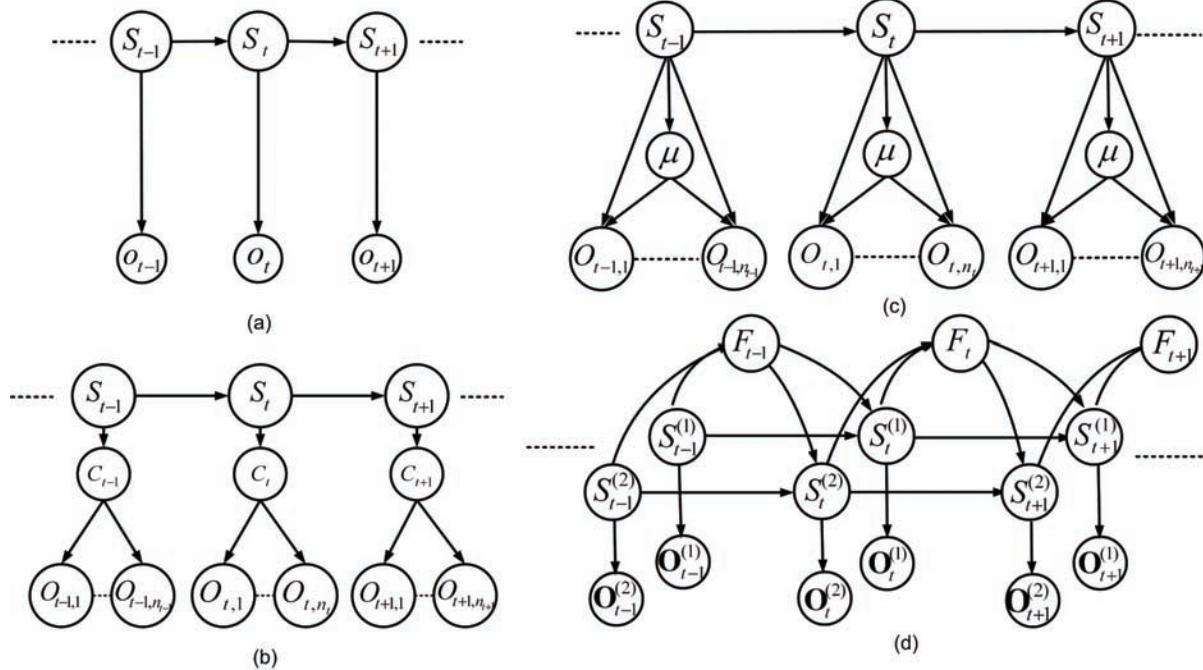
VISUAL FEATURES

As the first layer of the semantic space, relevant low-level visual features are needed to specify mid-level semantic structures, i.e., “camera views” and “play types”. Due to their different natures we need two complete different sets of visual features.

In Figure 3(a), we illustrate typical scenes for four different camera views which demonstrate the visual distinction between them. We identify the spatial color distribution and the angle of yard lines to be relevant visual features. We use the dominant color to estimate the spatial color distribution, and the Robust Dominant Color Region Detection algorithm (Ekin, Tekalp, & Mehrotra, 2003) to extract the dominant color region: the play ground. In order to estimate the color distribution, we segment a frame into three regions horizontally and three regions vertically, as shown in Figure 3. Each region occupies a 1/3 area of the frame. To obtain the yard line angles, we use Canny edge detection and the Hough transform to detect the yard lines. Based on the detected playing field and yard lines, we extract a 6-D feature vector composed by the following features: (1) the ratio of the dominant color region; (2) the ratio difference of dominant color between the left/right and center parts; (3) the ratio difference of dominant color between the left and right parts; (4) the ratio difference of dominant color between the top and bottom parts; (5) the average angle of all yard lines; (6) the angle difference between the first and last frames in a shot (Ding & Fan, 2006).

Play types are largely dependent on the pattern of camera motion (Lazarescu & Venkatesh, 2003). There are mainly two types of camera motion: panning and tilting, as shown in Figure 3(b). Most play types can be effectively characterized by these two kinds of camera motion. For example, in a long run from right to left, camera motion could be a short right-panning then a long left-panning, which is dif-

Figure 4. Four HMMs: (a) the traditional HMM, (b) the embedded HMM-GMM, (c) the segmental HMM (SHMM), (d) the Multi-channel segmental HMM (MCSHMM).



ferent from the motion trend of a short run, i.e., a short right-panning followed by a short left-panning. To specify camera motion we choose the optical flow based method (Srinivasan, Venkatesh, & Hosie, 1997) to qualitatively compute parameters of the panning and tilting between two adjacent frames, e.g., *panning* and *tilting*. Also, the frame indices are included as an additional temporal feature to distinguish long plays and short plays. Therefore, we compose a 3-D feature vector of camera motion for every two adjacent frames in each shot, which will be employed to identify the play type in each shot.

HMM-BASED SEMANTIC VIDEO ANALYSIS

HMMs are the most popular machine learning tool for video mining. A typical HMM as shown in Figure 4 (a) is a statistical model where the underlying system is assumed to be a Markov process with unknown parameters including a state transition matrix and a probabilistic observation model. The former one captures the underlying state dynamics, and the latter one characterizes observations pertaining to the hidden states either as probability density functions (continues observations) or probability mass functions (discrete observations). There are three canonical problems about HMMs (Rabiner, 1989):

Evaluation: given the model parameters, compute the probability of a particular observation sequence. This problem is solved by the forward-backward algorithm.

Learning: given an observation sequence, discover the model parameters that include the set of state transition and output probabilities. This problem is solved by the Baum-Welch algorithm, or the Expectation Maximization (EM) algorithm.

Decoding: given the model parameters, find the most likely sequence of hidden states that could have generated a given observation sequence. This problem is solved by the Viterbi algorithm.

In this work, we assume that a video sequence is composed of a set of pre-segmented consecutive shots each of which was captured from one of four camera views and corresponds to one of four play types. In the following, we will first introduce the basic HMM-based approach where we want to infer mid-level semantic structures from low-level visual features. This algorithm will serve as the reference for our future improvements. Specifically, we have three new HMM-based approaches which are called the embedded HMM-GMM, the segmental HMM (SHMM), and the multi-channel segmental HMM (MCSHMM) as shown in Figure 4(b)(c)(d), respectively.

Hidden Markov Models

We first discuss the implementation of a basic HMM for semantic video analysis that follows our problem formulation and provides a benchmark for our following studies. According to our problem formulation, we want to use the hidden state in the HMM to encode the mid-level semantic structure at the shot level directly. Correspondingly, we can define two HMMs. One is for camera view analysis where yard line angles and color distributions are used as the observation, and the other is for play type analysis where camera motion is used as the observation. It is worth noting that the hidden state is defined at the shot level, while the observations at the frame-level. Tradition HMMs cannot handle variable length observations, and an easy treatment to obtain the shot-wise feature is to compute the average of all frame-wise observations in a shot.

Given a video sequence of T shots, there are four latent states that correspond to four camera views or play types, i.e., $\{S_t = k \mid t = 1, \dots, T; k = 1, 2, 3, 4\}$. By the Markov assumption we can have a state transition matrix, i.e., $\{a_{k,j} = p(S_t = j \mid S_{t-1} = k) \mid k, j = 1, 2, 3, 4\}$, to govern the state dynamics over time as shown in Figure 1. Observations, $\{o_t \mid t = 1, \dots, T\}$, i.e., relevant visual features extracted from all shots, are assumed to be drawn from certain emission function associated with the latent states.

The often used emission functions include Gaussian or the Gaussian mixture model (GMM). Then we can define two kinds of observation models as

$$p(o_t \mid S_t = k) = N(o_t \mid \mu_k, \Sigma_k), \quad (1)$$

where μ_k and Σ_k are the mean and covariance matrix of the Gaussian for latent state k , or

$$p(o_t \mid S_t = k) = \sum_{n=1}^N \alpha_n N(o_t \mid \mu_{nk}, \Sigma_{nk}), \quad (2)$$

where $\{\alpha_n, \mu_{nk}, \Sigma_{nk} \mid n = 1, \dots, N\}$ parameterizes the N -order GMM for latent state k , and $\sum_{n=1}^N \alpha_n = 1$.

Given a series of T -shot observations $\{o_t \mid t = 1, \dots, T\}$, the HMM with GMM emission can be parameterized by $\Gamma = \{S, \pi_k, a_{k,j}, \alpha_n, \mu_{nk}, \Sigma_{nk} \mid k = 1, \dots, 4; n = 1, \dots, N\}$:

- $S = k, k \in \{1, 2, 3, 4\}$: the hidden states represent four camera views or play types;
- $\pi_k = p(S_1 = k)$: Initial state probabilities vector, which represents the initial probabilities of four camera views or four play types;
- $A = \{a_{k,j} = p(S_t = j | S_{t-1} = k) | t = 1, \dots, T; j, k = 1, 2, 3, 4\}$: are the state transition probabilities between states (camera views or play types) j and k ;
- $p(o_t | S_t = k) = \sum_{n=1}^N \alpha_n N(o_t | \mu_{nk}, \Sigma_{nk})$: is the N -order GMM emission function of hidden states, which is reduced to the Gaussian emission with order one ($N=1$).

Then we can use the EM algorithm (Bilmes, 1997) to obtain the maximum likelihood-based parameter estimate of this HMM as follows:

$$\Gamma^* = \arg \max_{\Gamma} p(o_{1:T} | \Gamma), \quad (3)$$

where

$$p(o_{1:T} | \Gamma) = \sum_k \pi_k \prod_{t=1}^T p(S_{t+1} | S_t) p(o_t | S_t = k). \quad (4)$$

After the EM training, we can use the Viterbi decoding algorithm (Bilmes, 1997) to estimate the optimal state sequence

$$S_{1:T}^* = \arg \max_{S_{1:T}} P(S_{1:T} | o_{1:T}, \Gamma^*), \quad (5)$$

which corresponds to the camera view or play type classification results for all T shots. Thus the proposed HMM-based video representation framework can explicitly explore semantic structures in American football videos. However, the major limitation is that each shot is represented by an averaged feature vector that is less representative and informative. This fact motivates us to improve the observation model in the HMM.

Embedded HMM-GMM

As mentioned before, the latent state, i.e., the camera views and the play types, are defined at the shot level while the observations are extracted from all frames in a shot. Averaging frame-wise features in one shot to obtain the shot-wise feature inevitably reduce the discriminability of observations in the HMM. Also, each shot may have different lengths, i.e., the number of frames varies from shot to shot. In order to take advantage of the rich statistics of frame-wise visual features and to handle variable-length observations, we have proposed a new HMM that is embedded with a two-layer observation model, as shown in Figure 4(b) (Ding & Fan, 2007b).

In the new model, we assume there is a virtual observation layer between shot-wise latent states and frame-wise observations that is represented by a GMM. Unlike the traditional HMM where each state

directly emits an observation, the state in the new HMM is associated with an M -order Gaussian Mixture Model (GMM). Then there are two steps in the process of observation generation for shot t . First, given a state, a GMM is drawn and then a set of frame-wise observations $\mathbf{O}_t = \{o_{t,i} \mid i = 1, \dots, n_t\}$ are drawn from this GMM where n_t is the number of frames in shot t . Therefore, we can define the emission function of virtual observation as:

$$p(o_{t,i} \mid \Pi_k, S_t = k) = \sum_{m=1}^M \alpha_m^k N(o_{t,i} \mid \mu_m^k, \Sigma_m^k), \quad (6)$$

where $\Pi_k = \{\alpha_m^k, \mu_m^k, \Sigma_m^k\}$ is the parameter set of the M -order GMM ($M=3$ in this work) for state k and $\sum_{m=1}^M \alpha_m^k = 1$. To accommodate variable-length observations we can define a normalized probability density function of the frame-wise observations as

$$\log p(\mathbf{O}_t \mid S_t = k) = \frac{1}{n_t} \sum_{i=1}^{n_t} \log p(o_{t,i} \mid \Pi_k, S_t = k) \quad (7)$$

The proposed model extends the traditional HMM where the state is defined on each observation. In the new model the GMM is built in the HMM as an additional observation layer in order to utilize the rich statistics of frame-wise observations. Therefore, we call this new model the *embedded HMM-GMM*. It is worth mentioning that it is different from the traditional HMM with GMM emission where the state directly emits an observation, while in the new model the state emits a GMM that further emits a set of observations.

Given a series of observations of T shots, $\mathbf{O}_{1:T} = \{\mathbf{O}_t \mid t = 1, \dots, T\}$, the likelihood function the proposed model is defined as:

$$p(\mathbf{O}_{1:T} \mid \Gamma) = \sum_k \pi_k \prod_{t=1}^T p(S_{t+1} \mid S_t) p(\mathbf{O}_t \mid S_t = k), \quad (8)$$

where $\Gamma = \{\pi_k, a_{k,j}, \Pi_k \mid k, j = 1, \dots, 4\}$. Specifically, we define the posterior probability of each shot and the joint probability between two adjacent shots as:

$$\gamma_k(t) = p(S_t = k \mid \mathbf{O}_{1:T}, \Gamma) \quad \text{and} \quad \xi_{k,j}(t) = p(S_t = k, S_{t+1} = j \mid \mathbf{O}_{1:T}, \Gamma). \quad (9)$$

Due to the two-layer observation model, estimating Γ requires updating Π_k that involves multiple GMMs to characterize observations across shots. Moreover, in order to avoid the training bias due to variable-lengths of different shots we use a Monte Carlo-like sampling method to construct a fixed size training pool for K GMMs by sampling from \mathbf{O} . Then the training data pools for all states are represented by

Table 1. Algorithms I-III

Algorithm I: Constructing the training pools for GMMs associated with K states Input: Given observations of T shots, set the minimum sampling number for each shot is B . Output: New training data pool $\mathbf{X}^{\{1:k\}}$ for KGMMs For $k=1$ to K do For $t=1$ to T do If $n_t > B$ then If $n_t * \gamma_t > B$ Then $W = B$ Else $W = B * \gamma_k(t)$; Else if $n_t \leq B$ then: If $n_t > \gamma_t * B$ Then $W = B * \gamma_k(t)$ Else $W = B$ Uniformly sample W feature vectors in shot t End Obtain training data pool, $\mathbf{X}^{\{k\}}$ End	Algorithm II: EM for the GMM-based first-layer observation model Input: Set initial parameter set Π_0 and the iteration number I , and given the training data $\mathbf{X}^{\{1:k\}}$ from Algorithm I Output: New parameter set $\hat{\Pi}$ For $k=1$ to K do For $i=1$ to I do E-step: compute $p(m x_{t,w}^k, \Pi_k)$ (10) M-step: compute α_m^k (11) compute μ_m^k (12) compute Σ_m^k (13) End End	Algorithm III: EM for the Embedded HMM-GMM Input: Set initial parameter set Γ_0 and the iteration number J . Output: New parameter set $\hat{\Gamma}$ For $j=1$ to J do E-step: Compute $\gamma_k(t)$ Compute $\xi_{k,j}(t)$ (9) M-step: compute $\hat{\pi}_k$ compute $\hat{a}_{k,j}(t)$ (14) compute $\hat{\alpha}_m^k, \hat{\mu}_m^k, \hat{\Sigma}_m^k$ (Algorithm II); End
--	---	---

$$\mathbf{X}^{\{1:k\}} = \{x_{t,w}^k \mid k = 1, \dots, K; t = 1, \dots, T; w = 1, \dots, W\},$$

where W represents the sampling number in each shot.

After sampling, we can update the GMM for each hidden state. The likelihood of $x_{t,w}^k$ with respect to k , i.e., the GMM of state k , is computed by

$$p(m | x_{t,w}^k, \Pi_k) = \frac{\alpha_m^k p_m(x_{t,w}^k | \Pi_k)}{\sum_m \alpha_m^k p_m(x_{t,w}^k | \Pi_k)} \quad (10)$$

and Π_k can be updated as following:

$$\alpha_m^k = \frac{1}{|X^k|} \sum_t \sum_w p(m | x_{t,w}^k, \Pi_k), \quad (11)$$

$$\mu_m^k = \frac{\sum_t \sum_w x_{t,w}^k p(m | x_{t,w}^k, \Pi_k)}{\sum_t \sum_w p(m | x_{t,w}^k, \Pi_k)}, \quad (12)$$

$$\Sigma_m^k = \frac{\sum_t \sum_w x_{t,w}^k p(m | x_{t,w}^k, \Pi_k) (x_{t,w}^k - \mu_m^k) (x_{t,w}^k - \mu_m^k)^T}{\sum_t \sum_w p(m | x_{t,w}^k, \Pi_k)}. \quad (13)$$

Finally, Γ is obtained by maximizing the expectation of the likelihood of the virtual observation considered, as

$$\pi_k = \gamma_k(1) \quad \text{and} \quad a_{k,j}(t) = \frac{\sum_t \xi_{k,j}(t)}{\sum_t \gamma_k(t)} \quad (14)$$

Essentially, the training of HMM-GMM involves two EM algorithms. One for the GMM-based first-layer observation model and the other is for the embedded HMM-GMM. The first one is embedded in the second one. Also, the first one involves a sampling process to construct the training pool for GMMs associated with K states. The full learning algorithm of the embedded HMM-GMM is shown in Table 1.

After the EM training, we can use the Viterbi decoding algorithm to estimate the optimal state sequence that corresponds to the play types or the camera views of T shots. The key idea in the embedded HMM-GMM is the two-layer observation model that involves GMMs as the first-layer shot-wise observation and the visual features as the second-layer frame-wise observation. This two-layer observation model is able to accommodate variable-length observations and to (partially) take advantage of rich statistics of frame-wise features for state inference.

However, there are still two limitations in the embedded HMM-GMM. One is that it assumes that frame-wise observations in one shot are independent, and other is that the learning of first-layer GMMs does not fully use all available frame-wise features due to the sampling method that is used to avoid the training bias introduced by variable-length shots. Still the promising results from this method motivate us to further improve the observation model in the HMM with better generality and capability.

Segmental HMM (SHMM)

Although the embedded HMM-GMM is able to provide a more effective observation model compared with the traditional HMM, it still assumes that all frame-wise features are independent in a shot. This assumption ignores the temporal dependency across frames in a shot. Also, it only uses part of observations for the training purpose. Therefore, we invoke a segmental HMM (SHMM) to attack this problem that is able to handle variable-length observations in a more analytical way (Ding & Fan, 2007a).

The SHMM was first introduced for speech recognition that involves a segmental observation model, as shown in Figure 4(c) (Gales & Young, 1993). Instead of generating one observation by each hidden state in the traditional HMM (Figure 4(a)), each hidden state of the SHMM can emit a sequence of observations, which can be called a segment. In SHMM all observations in a given segment are assumed to be independent to the observations belonging to other segments. In addition, in each segment all observations are conditionally independent given the mean of that segment. Thus a closed form likelihood function is attainable that can capture rich statistics of frame-wise features in a shot and is able to accommodate the variable-length shots.

The SHMM can be characterized by $\Gamma = \{\pi_k, a_{k,j}, \mu_{\mu,k}, \Sigma_{\mu,k}, \Sigma_k \mid k, j = 1, \dots, K\}$. Given a segment of state k , we use a Gaussian $N(\mu \mid \mu_{\mu,k}, \Sigma_{\mu,k})$, to characterize the distribution of its mean μ , and we use another Gaussian function $N(o \mid \mu, \Sigma_k)$ to represent all observations in the segment, i.e., $\mathbf{O}_{1:T} = \{\mathbf{O}_t \mid t = 1, \dots, T\}$ and $\mathbf{O}_t = \{o_{t,i} \mid i = 1, \dots, n_t\}$, given its mean μ . Moreover, it is assumed that observations in a shot are

conditionally independent given the mean of that shot. Therefore, we can define the likelihood function of observations in shot t as follows:

$$p(\mathbf{O}_t | S_t = k, \Gamma) = \int p(\mu | S_t = k, \Gamma) \prod_{i=1}^{n_t} p(o_{t,i} | \mu, S_t = k, \Gamma) d\mu. \quad (15)$$

where

$$p(\mu | S_t = k, \Gamma) = N(\mu | \mu_{\mu,k}, \Sigma_{\mu,k}) \quad (16)$$

and

$$p(o_{t,i} | \mu, S_t = k, \Gamma) = N(o_{t,i} | \mu, \Sigma_k). \quad (17)$$

Essentially, the observation model in the SHMM involves a two-layer observation model where two Gaussians defined in (16) and (17) are used to capture the *inter-segment* and *intra-segment* feature variability respectively. It can be handily extended to semantic video analysis (Ding & Fan, 2007a), where the two Gaussians can represent the *inter-shot* and *intra-shot* variability of observations. In speech recognition, the length of each segment is unknown that has to be estimated during the EM algorithm. While in this work, the length of each video shot is known, making the EM learning much simpler. Compared with the traditional HMM, only one additional parameter Σ_k is added to each state in the SHMM. Compared with the embedded HMM-GMM, the SHMM also has a two-layer observation model that is able to effectively capture the temporal dependency of frame-wise observations in one shot by two Gaussian functions.

To simplify the likelihood function defined in (15), we use the following definitions:

$$\mu_t = \sum_{i=1}^{n_t} o_{t,i} \quad (18)$$

$$\Sigma_{t,k} = (\Sigma_{\mu,k}^{-1} + n_t \Sigma_k^{-1})^{-1} \quad (19)$$

$$\mu_{t,k} = (\Sigma_{t,k} (\Sigma_{\mu,k}^{-1} \mu_{\mu,k} + \Sigma_k^{-1} \mu_i))' \quad (20)$$

Then we can rewrite (15) by eliminating the integration as

$$\log p(\mathbf{O}_t | S_t = k, \Gamma) = \log(R_k R_{\mu,k} R_{t,k}) + \frac{1}{2} \mu_{\mu,k} \Sigma_{\mu,k} (\mu_{\mu,k})' + \frac{1}{2} \sum_{i=1}^{n_t} o_{t,i} \Sigma_k^{-1} (o_{t,i})' - \frac{1}{2} \mu_{t,k} \Sigma_{\mu,k}^{-1} (\mu_{t,k})', \quad (21)$$

$$\text{where } R_k = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma_k|^{\frac{1}{2}}}, R_{\mu,k} = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma_{\mu,k}|^{\frac{1}{2}}}, R_{t,k} = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma_{t,k}|^{\frac{1}{2}}}.$$

Then we can employ the Backward-Forward algorithm to calculate the total probability summed over all possible paths.

$$\alpha_k(t) = p(\mathbf{O}_1, \dots, \mathbf{O}_t, S_t = k | \Theta) \quad (22)$$

$$\beta_k(t) = p(\mathbf{O}_{t+1}, \dots, \mathbf{O}_T, S_t = k | \Theta) \quad (23)$$

$$\gamma_k(t) = \frac{\alpha_k(t)\beta_k(t)}{\sum_k \alpha_k(t)\beta_k(t)} \quad (24)$$

$$\xi_{k,j}(t) = \frac{\gamma_k(t)a_{k,j}p(\mathbf{O}_{t+1} | S_{t+1}, \Theta)\beta_j(t+1)}{\beta_k(t)} \quad (25)$$

Finally, the parameter set of the SHMM can be estimated as follows:

$$\hat{\mu}_{\mu,k} = \frac{\sum_{t=1}^T \gamma_k(t) \frac{\mu_t}{n_t}}{\sum_{t=1}^T \gamma_k(t)} \quad (26)$$

$$\hat{\Sigma}_{\mu,k} = \frac{\sum_{t=1}^T \gamma_k(t) \frac{(n_t \hat{\mu}_{\mu,k} - \mu_t)^2}{n_t^2}}{\sum_{t=1}^T \gamma_k(t)} \quad (27)$$

$$\hat{\Sigma}_k = \frac{\sum_{t=1}^T \gamma_k(t) (\sum_{i=1}^{n_t} (o_{t,i})^2 - \frac{\mu_t^2}{n_t})}{\sum_{t=1}^T \gamma_k(t) (n_t - 1)} \quad (28)$$

$$\hat{a}_{k,j} = \frac{\xi_{k,j}(t)}{\gamma_k(t)} \quad (29)$$

$$\hat{\pi}_k = \gamma_k(1) \quad (30)$$

After the EM training, we can use the Viterbi algorithm to estimate the optimal state sequence $S_{1:T}^*$ as before. More details can be found from (Ding & Fan, 2007a).

Multi-Channel Segmental HMM (MCSHMM)

All HMM-based approaches we discussed so far can only detect one semantic structure at one time. Usually, a sports video is featured by the co-existence of multiple mid-level semantic structures such as play types and camera views. More interestingly, there are some interactions among them. It was observed that camera views and play types are quite related to each other during a football game. For example, after a shot of the central view accompanied by a short play, the camera view in the next shot is likely to remain in the same view; while if it is a long play, the next camera view might be switched to the other camera views. Therefore, we are interested in exploring the interaction between multiple semantic structures with the aim to improve the overall mining performance.

It is tempting for us to use the Coupled HMM (CHMM) proposed in (M. Brand, et al., 1997) to explore the interaction between multiple Markov channels. The CHMM involves two parallel Markov processes that are defined in the same state space and share the same semantics, although observations could come from different sources or modalities. More importantly, the CHMM usually assume relatively strong interaction between two Markov chains that may overweight or even corrupt the Markovian property within each individual chain if the assumption is not true, as observed in our practice.

Therefore, we advance a new multi-channel SHMM (MCSHMM) that involves two *parallel* SHMMs and a two layer *hierarchical* Markovian structure, as shown in Figure 4(d). In the proposed model, on the one hand, the SHMM based observation model can represent rich variable-length observations for each individual semantic structure. On the other hand, the MCSHMM can explore the interaction between multiple semantic structures via a parallel-hierarchical dynamic model that is able to balance the statistical dependency both within each Markov process and across two Markov processes. By incorporating these two aspects into one framework, we are able to mine two mid-level semantic structures simultaneously (Ding & Fan, 2008).

The unique feature of the MCSHMM is the integration of both parallel and hierarchical structures on latent states and the incorporation of a segmental observation model, which enable the MCSHMM to have more flexibility, capacity, and functionality than the existing HMMs, including CHMM, SHMM and HHMM etc. In the view of DBN, both the dynamic model (among hidden states) and the observation model in the MCSHMM have a two-layered structure that greatly enhances its capability and flexibility of learning and inference.

Specifically, at the first-layer of the dynamic model, $\mathbf{S} = \{S_t^{(j)} | t = 1, \dots, T; j = 1, 2\}$ denotes the state sequence of two channels where $S_t^{(j)}$ denotes the state of shot t in channel j , and at the second-layer of the dynamic model, $\mathbf{F} = \{F_t = (S_t^{(1)}, S_t^{(2)}) | t = 1, \dots, T\}$ represents the state sequence at the second layer where each state consists of the states at the first layer. At the observation layer, $\mathbf{O} = \{o_{t,i}^{(j)} | t = 1, \dots, T; i = 1, \dots, n_t; j = 1, 2\}$ indicates observations of shot t of n_t frames in two channels.

Therefore, the parameter set of MCSHMM is represented by $\Gamma = \{\mathbf{A}, \Pi, \Omega\}$ that includes the following parameters.

- *Initial probabilities:* $\Pi = \{P(S_1^{(1)}), P(S_1^{(2)}), P(F_1 | S_1^{(1)}, S_1^{(2)})\};$
- *Transition probabilities:* $\mathbf{A} = \{A_w, w = 1, 2, 3\},$

where

$$\begin{aligned} A_1 &= \{P(S_t^{(1)} = m \mid S_{t-1}^{(1)} = n, F_{t-1} = l) \mid m, n = 1, \dots, 4, l = 1, \dots, 16\}, \\ A_2 &= \{P(S_t^{(2)} = m \mid S_{t-1}^{(2)} = n, F_{t-1} = l) \mid m, n = 1, \dots, 4, l = 1, \dots, 16\}, \\ A_3 &= \{P(F_t = l \mid S_t^{(1)} = m, S_t^{(2)} = n) \mid m, n = 1, \dots, 4, l = 1, \dots, 16\}; \end{aligned}$$

- Observation density functions:

$$p(\mathbf{O}_t^{(j)} \mid S_i^{(j)} = m, \Omega) = \int N(\mu \mid \mu_{\mu,m}^{(j)}, \Sigma_{\mu,m}^{(j)}) \prod_{i=1}^{n_t} N(o_{t,i}^{(j)} \mid \mu, \Sigma_m^{(j)}) d\mu, \quad (31)$$

where $\Omega = \{\mu_{\mu,m}^{(j)}, \Sigma_{\mu,m}^{(j)}, \Sigma_m^{(j)} \mid j = 1, 2; m = 1, \dots, 4\}$ specifies two segmental models, and $o_{t,i}^{(j)}$ denotes the observation from the i th frame in shot t of channel j . Then, given the dual-channel observations \mathbf{O} of T shots, the joint likelihood is defined as

$$\begin{aligned} p(\mathbf{S}, \mathbf{F}, \mathbf{O} \mid \Gamma) &= P(S_1^{(1)})P(S_1^{(2)})P(F_1 \mid S_1^{(1)}, S_1^{(2)}) \\ &\quad \prod_{t=1}^T P(F_t \mid S_t^{(1)}, S_t^{(2)}) \prod_{t=2}^T \prod_{j=1}^2 P(S_t^{(j)} \mid S_{t-1}^{(j)}, F_{t-1}) \prod_{t=1}^T \prod_{j=1}^2 p(\mathbf{O}_{t,i}^{(j)} \mid S_t^{(j)}) \end{aligned} \quad (32)$$

Although the MCSHMM has some advantages in terms of its structure, the model learning is challenging and even problematic due to a large state space involved in the two-layer dynamic model represented by \mathbf{F} and \mathbf{A} . As we mentioned before there are two aspects in model learning, *structure learning* and *parameter learning*. The former one aims at finding a compact and effective model structure by simplifying the state space and reducing the parameter set, and the latter one tries to optimize the model parameters given a model structure. Recent studies show that two issues could be jointly formulated and optimized in one learning framework for optimal performance (Binsztok & Artières, 2004; Freitag & Mccallum, 2000).

Inspired by the ideas of entropic prior and parameter extinction (Matthew Brand, 1999), we propose a new unsupervised learning algorithm framework where the structure and parameters of the MCSHMM can be optimized simultaneously. Essentially, it results in a maximum *a posteriori* (MAP) estimator that encourages a maximally structured and minimally ambiguous model. This is accomplished by trimming the weakly supported parameters and states, leading to a compact and concise model with good determinism. We first initialize the model structure of MCSHMM to cover all possible configurations in the state space. This MAP estimator is incorporated in the Maximization-step of the EM algorithm to optimize the model structure and parameters.

The MAP estimator mainly focuses on transition matrices \mathbf{A} that have many possible state transitions due to a large number of possible states in the second-layer, i.e., \mathbf{F} (16 in this work). In other words, we want to simplify \mathbf{A} by keeping only important state transitions, which would effectively reduce the number of useful states in \mathbf{F} and balance the incoming and outgoing transitions between the two layers. Consequentially, the MAP-based EM estimator finds the optimal parameter by

$$\Gamma^* = \arg \max_{\Gamma} p_e(\mathbf{O} \mid \Gamma) \quad (33)$$

where

$$p_e(\mathbf{O} \mid \Gamma) \propto p(\mathbf{O} \mid \Gamma)p_e(\Gamma) = p_e(\Gamma) \sum_{S,F} p(\mathbf{S}, \mathbf{F}, \mathbf{O} \mid \Gamma) \quad (34)$$

and $p_e(\Gamma)$ is the entropic prior of the model corresponding to parameter set Γ that, in this work, depends on \mathbf{A} as

$$p_e(\Gamma) \propto \exp\left(\sum_w \sum_p \sum_q P_{p,q}^w \log P_{p,q}^w\right), \quad (35)$$

where $P_{p,q}^w$ denotes a transition probability in A_w . Accordingly, in the M-step of the EM algorithm, we will update the transition probabilities by maximizing the entropic prior as

$$\mathbf{A}^* = \arg \max_{\mathbf{A}} \{(\log p_e(\mathbf{O} \mid \Gamma)) + \sum_w \sum_p \lambda_p^w (\sum_q P_{p,q}^w - 1)\}, \quad (36)$$

where λ is the Lagrange multiplier to ensure $\sum_q P_{p,q}^w = 1$.

Consequentially, this step will enhance the important states in \mathbf{F} and drive the weakly supported ones towards zero. According to the transitions of small probabilities, the states in \mathbf{F} that are seldom visited could be found and eventually eliminated. Hereby the state space is optimized by balancing the state transitions between the two layers i.e., $\{S_t^{(1)}, S_t^{(2)} \leftrightarrow F_t\}$ that is represented by \mathbf{A} . After \mathbf{A} is optimized, other parameters can also be updated in the M-step accordingly.

We resort to the junction tree algorithm in (Murphy, 2002) to implement the MAP-based EM algorithm for the MCSHMM. The junction tree is an auxiliary data structure that can convert a Directed Acyclic Graph (DAG), such as the MCSHMM, into an undirected graph by eliminating cycles in the graph, so that belief propagation can be effectively performed on the modified graph for inference and learning. Our EM algorithm implementation was based on the Bayes Net Matlab Toolbox developed by KP Murphy¹.

After EM learning, the second layer state set \mathbf{F} is available that captures the dependency between two channels. In this work, it is trimmed to a set of six states that is less than 40% of the complete state set that is 16. The Viterbi algorithm can be used to obtain optimal state sequences for both channels at the first layer, i.e., $\{S_t^{(j)} \mid t = 1, \dots, T; j = 1, 2\}$, which encodes two semantic structures, i.e., camera views and play types.

Experimental Results on Four HMMs

Our experiments were based on eight 30-minute football games recorded from the broadcasting NFL program. The video data is recorded as the MPEG-1 video stream (352×240 pixels, 30fps). All video shots have been manually labeled with two mid-level semantic structures that will be used as the ground-truth for algorithm validation. We have extracted two types of visual features for camera views and play

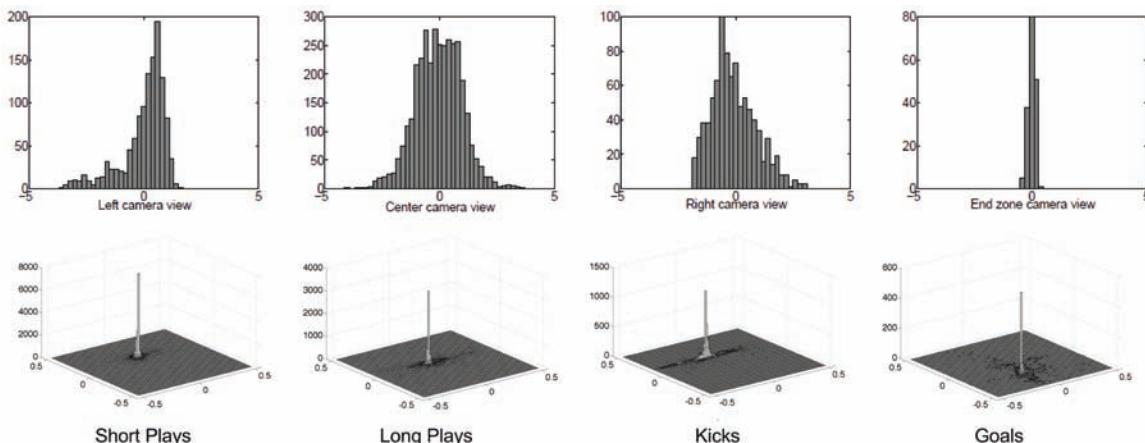
types, respectively. We compute both the frame-wise feature and the shot-wise feature that is obtained by averaging frame-wise features in each shot. Totally, we have evaluated seven HMM-based algorithms all of which were developed in Matlab 7.0 and tested on a desktop with 3.2GHz CPU and 1GB memory. They are the HMM with Gaussian emission ($HMM^{(1)}$) and GMM emission ($HMM^{(2)}$) (Ding & Fan, 2006), and the embedded HMM-GMM ($HMM^{(3)}$) (Ding & Fan, 2007b), the SHMM (Ding & Fan, 2007a), and the MCSHMM (Ding & Fan, 2008). We also implemented two CHMM-based approaches (M. Brand, et al., 1997), among which CHMM⁽¹⁾ uses shot-wise observations like the HMM⁽¹⁾, and CHMM⁽²⁾ uses frame-wise observations like the SHMM.

Normally, the EM algorithm is an adequate optimization approach provided that the data (e.g., visual features) indeed follow Gaussian distributions. To validate this assumption, we perform the normality test by using the ground truth classification results regarding camera views and play types. As shown in Figure 5, we demonstrate the distributions of the average yard line angle given different camera views as well as those of camera motion (panning/tilting) for four different play types. It is shown these distributions indeed follow either the Gaussian or mixture Gaussian distribution, which assures the appropriateness and effectiveness of the EM algorithm for model learning.

All seven HMMs are learned by the EM algorithm. It is well known that the EM algorithm is only capable of finding the local optimum. Therefore, the model initialization is critical to the learning performance. Practically, we adopt a coarse-to-fine progressive initialization strategy that uses the training result of a simpler model to initialize a more complex model. Specially, we first use the K-mean algorithm to obtain the coarsest classification results, in which we choose the K value according to our specification of the hidden state space where we assume four states for play types or camera views. Then K-mean results are used to initialize HMM⁽¹⁾ whose training result was used to initialize HMM⁽²⁾, and so on for the HMM⁽³⁾ and the SHMM. Finally, the training result of SHMM was used to initialize the MCSHMM.

Additionally, some prior knowledge can also be used to initialize the EM algorithm. For example, the first shot is always in the central view with a kick off play, and we can initialize the state probabili-

Figure 5. The normality test of visual features used in this work. The distributions of the average yard line angle in different camera views (top) and those of camera motion (panning/tilting) in different play types (bottom).



ties of the first shot in the HMMs accordingly. Also, the camera view is likely to stay in the same view from shot to shot, which implies that the transition matrix of camera view has relatively large diagonal probabilities. Furthermore, the initialization of transition matrix may also require the computation of the frequencies of different state transitions in a couple of real games based on the ground-truth labels.

In addition, we have also tested the classification results for each individual camera view and play type. In Table 3, we show the results from two football games, and we compare the performance of SHMM and MCSHMM. We can clearly see the MCSHMM does improve the classification result almost for all camera views and play types (only for three out of 16 cases MCSHMM has the same result as SHMM).

In this work, we are interested in two mid-level semantic structures of American football videos, *camera views* and *play types*. Among seven HMM-based algorithms, four of them (HMM⁽¹⁾, HMM⁽²⁾, HMM⁽³⁾, SHMM) explore the two semantic structures individually and independently, and three of them (CHMM⁽¹⁾, CHMM⁽²⁾, MCSHMM) can jointly optimize the two tasks. Table 3 shows the experimental results where the seven algorithms are evaluated in terms individual classification accuracy for camera views and play types as well as the overall classification accuracy for all eight videos. Specifically, we have following observation and discussion.

- Observation models:* In this work, we investigate four types of observation model including the Gaussian model in HMM⁽¹⁾, the GMM in HMM⁽²⁾, the two-layer GMM in HMM⁽³⁾, and the segmental model in the SHMM and MCSHMM. The major difference between these observation models lies in the capability of capturing the rich statistics of the frame-wise observations both in a shot and across different shots. Compared with others, the segmental model is able to effectively represent variable-length observations for state estimation, and it serves as the building block for the MCSHMM.

Table 2. Shot-based semantic structure classification for seven HMM-based algorithms.

Test Videos	Semantic Structures	HMM ⁽¹⁾	HMM ⁽²⁾	HMM ⁽³⁾	CHMM ⁽¹⁾	SHMM	CHMM ⁽²⁾	MCSHMM
Video 1 (156 shots)	Play Types	43.59%	60.90%	77.56%	41.03%	80.13%	78.85%	82.05%
	Camera Views	55.77%	72.44%	76.28%	58.33%	79.49%	81.41%	84.62%
Video 2 (163 shots)	Play Types	49.07%	64.34%	70.56%	53.73%	77.91%	76.69%	81.59%
	Camera Views	61.35%	67.48%	79.14%	65.03%	74.05%	80.98%	85.28%
Video 3 (167 shots)	Play Types	44.91%	49.10%	58.08%	52.10%	68.86%	70.06%	75.45%
	Camera Views	53.29%	58.68%	61.08%	58.08%	74.85%	77.25%	81.44%
Video 4 (168 shots)	Play Types	58.33%	64.67%	70.66%	67.86%	77.98%	69.62%	82.74%
	Camera Views	64.88%	70.24%	73.21%	69.05%	79.17%	75.60%	84.52%
Video 5 (168 shots)	Play Types	50.59%	61.90%	72.02%	70.24%	74.40%	70.83%	80.95%
	Camera Views	56.55%	65.48%	73.81%	60.71%	73.21%	64.88%	77.38%
Video 6 (170 shots)	Play Types	64.70%	71.17%	70.59%	72.35%	75.88%	76.47%	83.53%
	Camera Views	68.24%	72.35%	75.29%	67.06%	81.18%	71.46%	87.06%
Video 7 (171 shots)	Play Types	56.14%	65.50%	72.51%	68.82%	78.95%	82.94%	84.80%
	Camera Views	62.57%	75.44%	77.78%	76.02%	80.11%	81.18%	88.30%
Video 8 (173 shots)	Play Types	63.01%	67.05%	69.36%	67.63%	75.14%	69.36%	82.08%
	Camera Views	66.07%	68.21%	71.68%	69.94%	77.46%	73.41%	84.97%
Average		57.44%	62.62%	71.85%	63.54%	77.42%	75.08%	82.92%

Table 3. Detailed comparison between MCSHMM and SHMM regarding the classification results for each individual camera views and play types in two test videos.

Approaches	Test videos	Central view	Left view	Right view	End zone view	Short play	Long play	Kick/punt	Field Goal
SHMM	Video 1	80.00%	79.79%	61.54%	100.00%	85.19%	71.43%	77.27%	81.82%
	Video 2	70.65%	77.78%	73.33%	81.82%	85.37%	65.91%	66.67%	92.31%
MCSHMM	Video 1	85.00%	84.04%	76.92%	100.00%	86.42%	71.43%	81.82%	90.91%
	Video 2	83.70%	86.67%	80.00%	100.00%	86.59%	75.00%	66.67%	100.00%

- *Dynamic models:* The improvement of MCSHMM over HMMs ^(1,2,3), the SHMM and two CHMMs lies in the new parallel-hierarchical dynamic mechanism imposed on the hidden states. In the single channel models including HMM-based algorithms and the SHMM, we can only infer one kind of semantic structure each time. Although the CHMM has the capability of representing the coupling effect between two Markov chains, it imposes the same state space in two channels that implies strong interaction between two channels. This assumption is not true in our case as shown Table 3 where the CHMMs provide worse results than the single-channel HMMs. The MCSHMM can effectively capture the mutual interaction between two Markov chains by introducing the second-layer dynamics that is capable of balancing the dependency both within each channel and across two channels with the aim to boost the overall learning performance.
- *Model learning:* Simultaneous structure learning and parameter learning are crucial for the effective use of MCSHMM. The traditional ML estimator cannot fully take advantage of the MCSHMM because the untrimmed model structure has a complex state space that deteriorates the model's flexibility to adapt to the real video data with various underlying structures. The entropic prior based MAP estimator is capable of finding the optimal model structure and parameters jointly that best fit individual video sequences of different properties.
- *Computational cost:* The MCHSMM (without program optimization) has the highest computational load (about 50 minutes for one video), while other methods are between 2-20 minutes. This is mainly due to the large number of possible states at the second-layer that need to be evaluated for structure optimization and the complexity of the segmental observation model. One possible improvement is to introduce two observation models of different complexity. The simpler one can be used for structure learning, and the other for parameter learning.

CONCLUSION AND FUTURE WORK

In this chapter, we have discussed several HMM-based approaches for sports video mining. We advocate the concept of semantic space that supports explicit semantic modeling and video mining is formulated as an inference problem. The semantic space involves a three-layer structure, including low-level visual features, mid-level semantic structures and high-level semantics. Although this chapter is only focused on how to infer mid-level semantic structures from low-level features, the proposed framework will develop a foundation for high-level semantic analysis by providing a rich set of mid-level semantic structures. Specifically, we address several technical issues on how to improve the capability and capacity of HMM, and the highlight of this work is the new MCSHMM that shows significant advantages over existing HMMs for sports video mining, specially its capability of capturing the interaction between multiple

semantic structure and fully handle variable-length observations.

Our future work will be two-fold. One is to enrich the set of mid-level semantic structures that could support more informative semantic modeling. We are currently investigating to add “possession” and “play completion” as the two additional mid-level descriptors. The other is to explore high-level semantics by using mid-level semantic structures. New machine learning techniques are needed in order to support the inference at this level that has more diverse dependency structures between mid-level and high-level semantics. Our long-term goal is to develop a unified statistical machine learning framework for sports video mining that supports both structure-based and event-based semantic analysis and can also be applied to other similar video mining applications.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees whose comments help us improve this paper. This work is supported by the National Science Foundation (NSF) under Grant IIS-0347613.

REFERENCES

- Assfalg, J., Bertini, M., Colombo, C., Bimbo, A. d., & Nunziati, W. (2003). Semantic annotation of soccer videos: Automatic highlights identification. *Computer Vision and Image Understanding*, 92(2-3), 285–305. doi:10.1016/j.cviu.2003.06.004
- Bilmes, J. (1997). *A gentle tutorial on the EM algorithm and its application to Parameter estimation for Gaussian mixture and hidden Markov models* (ICSI-Report-97-021).
- Binsztok, H., & Artières, T. (2004). *Learning HMM structure for on-line handwriting modelization*. Paper presented at the IEEE International Workshop on Frontiers in Handwriting Recognition.
- Brand, M. (1999). Structure learning in conditional probability models via an entropic prior and parameter extinction. *Neural Computation*, 11(5), 1155–1182. doi:10.1162/089976699300016395
- Brand, M., Oliver, N., & Pentland, A. (1997). *Coupled hidden Markov models for complex action recognition*. Paper presented at the IEEE Int'l Conference on Computer Vision and Pattern Recognition.
- Calic, J., Campbell, N., Dasiopoulou, S., & Kompatsiaris, Y. (2005, December). *An overview of multimodal video representation for semantic analysis*. Paper presented at the IEEE European Workshop on the Integration of Knowledge, Semantics and Digital Media Technologies.
- Campbell, M. (Ed.). (2006). *IBM research TRECVID-2006 video retrieval system*.
- Chang, P., Han, M., & Gong, Y. (2002). *Highlight detection and classification of baseball game video with hidden Markov models*. Paper presented at the IEEE Int'l Conference on Image Processing, Rochester, NY.
- Chang, S. F., & Hsu, W. (Eds.). (2006). *Columbia University TRECVID-2006 video search and high-level feature extraction*.

- Ding, Y., & Fan, G. (2006). *Camera view-based American football video analysis*. Paper presented at the Eighth IEEE International Symposium on Multimedia.
- Ding, Y., & Fan, G. (2007a). *Segmental hidden Markov models for view-based sport video analysis*. Paper presented at the IEEE Int'l Conference on Computer Vision and Pattern Recognition.
- Ding, Y., & Fan, G. (2007b). *Two-layer generative models for sport video mining*. Paper presented at the IEEE International Conference on Multimedia and Expo.
- Ding, Y., & Fan, G. (2008). *Multi-channel segmental hidden Markov models for sports video mining*. Paper presented at the The ACM Multimedia Conference.
- Duan, L.-Y., Xu, M., Chua, T.-S., Tian, Q., & Xu, C.-S. (2003). *A mid-level representation framework for semantic sports video analysis*. Paper presented at the ACM Multimedia Conference.
- Ekin, A., Tekalp, A., & Mehrotra, R. (2003). Automatic soccer video analysis and summarization. *IEEE Transactions on Image Processing*, 12(7), 796–807. doi:10.1109/TIP.2003.812758
- Freitag, D., & McCallum, A. (2000). *Information extraction with HMM structures learned by stochastic optimization*. Paper presented at the The Seventeenth National Conference on Artificial Intelligence.
- Friedman, N., & Koller, D. (2003). Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1/2), 95–125. doi:10.1023/A:1020249912095
- Gales, M., & Young, S. (1993). *The theory of segmental hidden Markov models* (Tech. Rep. CUED/F-INFENG/TR 133), Cambridge University.
- Ghahramani, Z. (2002). Graphical models: Parameter learning. In M. A. Arbib (Ed.), *The Handbook of brain theory and neural networks*. Cambridge, MA: MIT Press.
- Kokaram, A., Rea, N., Dahyot, R., Tekalp, M., Bouthemy, P., & Gros, P. (2006). Browsing sports video: Trends in sports-related indexing and retrieval work. *IEEE Signal Processing Magazine*, 23(2), 47–58. doi:10.1109/MSP.2006.1621448
- Lafferty, J., McCallum, A., & Pereira, F. (2001). *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*. Paper presented at the Eighteenth International Conference on Machine Learning (ICML).
- Lazarescu, M., & Venkatesh, S. (2003, July). *Using camera motion to identify types of American football plays*. Paper presented at the 2003 International Conf. on Multimedia and Expo.
- Mei, T., Ma, Y., Zhou, H., Ma, W., & Zhang, H. (2005). *Sports video mining with Mosaic*. Paper presented at the 11th IEEE International Multimedia Modeling Conference.
- Murphy, K. (2002). *Dynamic Bayesian networks: Representation, inference and learning*. UC Berkeley.
- Nallapati, R. (2004). *Discriminative models for information retrieval*. Paper presented at the 27th annual international ACM SIGIR conference on Research and development in information retrieval, Sheffield, United Kingdom.

- Natsev, A., Naphade, M. R., & Smith, J. R. (2004). *Semantic representation, search and mining of multimedia content*. Paper presented at the ACM SIGKDD.
- Nefian, A. V., Liang, L., Pi, X., Liu, X., & Murphy, K. (2002). Dynamic Bayesian networks for audio-visual speech recognition. *EURASIP Journal on Applied Signal Processing*, (11), 1–15.
- Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), 257–286. doi:10.1109/5.18626
- Sadlier, D., & Connor, N. (2005). Event detection in field-sports video using audio-visual features and a support vector machine. *IEEE Trans. Circuits Syst. Video Technology*, 15(10), 1225–1233. doi:10.1109/TCSVT.2005.854237
- Srinivasan, M., Venkatesh, S., & Hosie, R. (1997). Qualitative estimation of camera motion parameters from video sequences. *Pattern Recognition*, 30, 593–606. doi:10.1016/S0031-3203(96)00106-9
- Wang, P., & Ji, Q. (2005). *Multi-view face tracking with factorial and switching HMM*. Paper presented at the IEEE Workshop on Applications of Computer Vision (WACV/MOTION05).
- Wang, T., Li, J., Diao, Q., Hu, W., Zhang, Y., & Dulong, C. (2006). *Semantic event detection using conditional random fields*. Paper presented at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
- Xie, L., Chang, S., Divakaran, A., & Sun, H. (2003). Unsupervised mining of staistical temporal structures in video. In D. D. A. Rosenfeld (Ed.), *Video mining*. Amsterdam: Kluwer Academi Publishers.
- Xiong, Z. Y., Zhou, X. S., Tian, Q., Rui, Y., & Huang, T. S. (2006). Semantic retrieval of video - review of research on video retrieval in meetings, movies and broadcast news, and sports. *IEEE Signal Processing Magazine*, 23(2), 18–27. doi:10.1109/MSP.2006.1621445
- Zhang, D., Gatica-Perez, D., Bengio, S., & Roy, D. (2005). *Learning influence among interacting Markov chains*. Paper presented at the NIPS.

KEY TERMS AND DEFINITIONS

Video Mining: the process of discovering knowledge, structures, patterns and events of interest in the video data.

Hidden Markov Models: a kind of statistical model in which the system being modeled is assumed to be a Markov process with unknown parameters.

Semantic Structures: an organization or a pattern that represents specific meaning.

Sports Video Analysis: a process to discover the meaningful structure or events of interest in the sports video.

Segmental HMMs: A special type of HMM with segmental observation model that can handle variable length observations.

EM Algorithm: a statistical estimation algorithm that can find maximum likelihood estimates of parameters in probabilistic models, where the model depends on unobserved latent variables.

Entropic Prior: a prior model that minimizes a type of mutual information between the data and

the parameters.

Model Learning: given the data, to learn a probabilistic model with optimized structures and parameters.

ENDNOTE

¹ <http://www.cs.ubc.ca/~murphyk/Software/BNT/bnt.html>

Chapter 23

A Survey of Bayesian Techniques in Computer Vision

José Blasco

Instituto Valenciano de Investigaciones Agrarias (IVIA), Spain

Nuria Aleixos

Polytechnic University of Valencia, Spain

Juan Gómez-Sanchis

University of Valencia, Spain

Juan F. Guerrero

University of Valencia, Spain

Enrique Moltó

Instituto Valenciano de Investigaciones Agrarias (IVIA), Spain

ABSTRACT

The Bayesian approach to classification is intended to solve questions concerning how to assign a class to an observed pattern using probability estimations. Red, green and blue (RGB) or hue, saturation and lightness (HSL) values of pixels in digital colour images can be considered as feature vectors to be classified, thus leading to Bayesian colour image segmentation. Bayesian classifiers are also used to sort objects but, in this case, reduction of the dimensionality of the feature vector is often required prior to the analysis. This chapter shows some applications of Bayesian learning techniques in computer vision in the agriculture and agri-food sectors. Inspection and classification of fruit and vegetables, robotics, insect identification and process automation are some of the examples shown. Problems related with the natural variability of colour, sizes and shapes of biological products, and natural illuminants are also discussed. Moreover, implementations that lead to real-time implementation are explained.

INTRODUCTION

Learning techniques can be employed to learn meaningful and complex relationships automatically in a set of training data, and to produce a generalisation of these relationships in order to infer interpretations for new, unseen test data (Mitchell et al., 1996). Statistical learning uses the statistical properties observed in a training set. As an example of this, Bayesian theory provides a probabilistic approach to inference, which proves successful both for segmentation of images and classification of objects in computer vision.

The Bayesian approach to classification is intended to solve questions concerning how to assign a class to an observed feature pattern using probability estimations. This means that this approach is aimed at estimating the probabilities of an observed pattern's belonging to each of some pre-defined classes in a classification problem, and then assigning the pattern to the class to which it is most likely to be a member. This set of probabilities, which henceforth are called *a posteriori* probabilities, is determined using the Bayes theorem, expressed in equation (1). The Bayes theorem calculates the *a posteriori* probability, $P(\Omega_i|x)$, that an observed pattern x , constituted by a series of j features ($x_1 \dots x_j$), belongs to class Ω_i , from the *a priori* probability of this class, $P(\Omega_i)$, and the conditional probabilities $P(x|\Omega_i)$, which are the probabilities of finding this pattern in class Ω_i . We will refer to this term as *conditional probabilities* for short.

$$P(\Omega_i|x) = \frac{P(x|\Omega_i)P(\Omega_i)}{P(x)} \quad (1)$$

$P(x)$ is the probability that a pattern x is present throughout the population data, and this probability can be determined from the total probability theorem as:

$$P(x) = \sum_{i=1}^N P(x|\Omega_i)P(\Omega_i) \quad (2)$$

The Bayes theory assumes that a pattern x , whose class is unknown, belongs to the class Ω_i , as follows:

$$x \in \Omega_i \Leftrightarrow \max_{r=i} \{P(\Omega_r|x)\} \quad (3)$$

From equation (2), we can consider that $P(x)$ is only a factor of scale for the *a posteriori* probabilities to be standardized between 0 and 1. This factor is essential for normalization of $P(\Omega_i|x)$ and for verification of probability axiomatic principles. However, a look at the numerator of the Bayes theorem (equation 1) is enough to deduce whether a pattern belongs to one class or another. Thus, it can be deduced that the condition shown in (3) is completely analogous to the condition described in (4), which simplifies the calculations required to take a decision.

$$x \in \Omega_i \Leftrightarrow \max_{r=i} \{P(x|\Omega_r)P(\Omega_r)\} \quad (4)$$

One immediate consequence that derives from condition (3) and one of the most remarkable properties of Bayesian classifiers is that they are optimal, in the sense that they minimize the expected misclassification rate (Ripley, 1996). Conditions (3) and (4) are not the only formal decision rules for a Bayesian classifier – there is another completely equivalent rule defined from what are called discriminant functions $f_i(x)$. These functions are also calculated from the *a priori* probabilities and the conditional *probabilities*, $P(x|\Omega_i)$. A discriminant function f_i that minimizes the error probability of classification can be calculated for each class Ω_i from the following set of equations:

$$f_i(x) = \ln [p(x|\Omega_i)] + \ln [P(\Omega_i)] \quad (5)$$

Once the values of the discriminant functions have been calculated for the observed pattern (which are sometimes called *scores*), the decision rule consists in assigning it to the class whose discriminant function has the highest value (*score*).

In any case, the probability distribution of patterns in each class $P(x|\Omega_i)$ and the *a priori* probability of classes $P(\Omega_i)$ have to be known in order to build the decision rule, but in many real-world problems they are unknown. In these cases, a labelled training set is used to estimate them. This training set is a dataset of observed patterns in which the class of each of the patterns is known. The way to estimate the above-mentioned probabilities is the basis of the different implementations of the Bayesian classifiers found in the literature.

If there is no other information, $P(\Omega_i)$ can be estimated from the frequencies of each of the classes within the training set. If n_i is the number of patterns that belong to the class Ω_i and n_{train} is the number of patterns contained in the training set, then we obtain:

$$P(\Omega_i) = \frac{n_i}{n_{train}} \quad (6)$$

But what most differentiate the different implementations is how to estimate the conditional probabilities from the patterns in the training set. Here follows some examples and discussion that appear in the literature.

Although in some control applications there are insufficient data to estimate the *conditional probabilities* (James, 1985), this is not the case when classifying colours of pixels in image analysis, because the amount of data provided by the images is usually enormous. In this kind of applications, frequency histograms are calculated for each class as a way to estimate the *conditional probabilities*. This method is especially useful in cases where the nature of the data does not fit common probability distributions. For instance, when working with images in the HSL colour space, the H coordinate is an angle and does not fit a normal (Gaussian) distribution (Marchant & Onyango, 2003). In this case, the range of variation of individual characteristics of each pattern in the training set is divided into cells of a certain size in order to build a multidimensional histogram that provides an estimate of the *conditional probabilities*. The major drawback of this method is that results depend on the level of detail, which means that results depend on choosing the correct size of the cells to build the histogram. If the size is too small, too many zero frequencies can be allocated in the histogram and it will be very difficult to determine probabilities. However, this implementation has proved to be effective in many works related to computer vision in

agriculture (Leemans, Magein & Destain, 1999; Marchant & Onyango, 2003; Onyango, Marchant & Zwiggelaar, 1997).

When $P(\Omega_i)$ are assumed to be normal distributions, calculations are greatly simplified. The generic equation of a multivariate normal distribution can be seen in equation (7), η_i being the mean of the patterns in class Ω_i and ζ_i the covariance matrix in this class. Both the mean and the covariance matrix can be estimated from the training set.

$$p(x|\Omega_i) = \frac{1}{\sqrt{(2\pi)^M |\zeta_i|}} \exp\left[-\frac{1}{2}(x - \eta_i)^T \zeta_i^{-1} (x - \eta_i)\right] \quad (7)$$

Using equation (7) and the expression of the discriminant functions shown in (5), discriminant functions f_i for the particular case of Gaussian distributions can be described as:

$$f_i(x) = -\frac{1}{2}\left[(x - \eta_i)^T \zeta_i^{-1} (x - \eta_i) + M \ln(2\pi) + \ln(|\zeta_i|)\right] + \ln(P(\Omega_i)) \quad (8)$$

When all the *conditional probabilities* are normal, the patterns are distributed into ellipsoids, whose sizes and orientations depend on the covariance matrices. The surfaces that separate the classes are quadratic functions and, for this reason, these classifiers are called quadratic or are based on Quadratic Discriminant Analysis.

When the covariance matrices of all classes are equal ($\zeta_i = \zeta$), the patterns of each class are distributed into ellipsoids of equal orientation and size whose centroids are η_i . In this case, classes are separated by hyperplanes and these classifiers are called linear or are based on Linear Discriminant Analysis (LDA). If *a priori* probabilities $P(\Omega_i)$ are also equal, the expression of the discriminant functions are then simplified to:

$$f_i(x) = -\frac{1}{2}\left[(x - \eta_i)^T \zeta^{-1} (x - \eta_i)\right] \quad (9)$$

Under these circumstances discriminant functions match the Mahalanobis distance of the observed pattern to the centroid of each class and Bayesian classifiers provide the same results as Fisher's Discriminant Analysis (Duda, Hart & Stork, 2001).

Red, green and blue (RGB) values of digital colour images can often be considered as normally distributed variables with equal variances in each case and, for this reason, colour segmentation of images can be implemented easily. Moreover, in these cases no regularity problems ($|\zeta_i|=0$) appear, which would make it impossible to calculate the inverse of the covariance matrices ζ_i^{-1} .

However, many applications require a great number of parameters to classify objects. In these cases, the covariance matrices have large dimensions, which produce regularity problems. Moreover, increasing the dimensionality of the problem may lead to inconsistent results, since the probability of correct classification of an object by chance increases as the ratio between the number of samples in the training set and the number of features decreases. For these reasons, in many classification problems a reduction of features is required prior to applying Bayesian analysis. One of the most extended methods for the selection of the features that have more discriminatory power is stepwise analysis.

A stepwise feature selection procedure begins with no variables in the classification model. At each step, variables (features) within and outside the model are evaluated. This evaluation consists in finding the variable within the model that makes the least contribution to its discriminatory performance, as determined by the Wilks' Lambda method, and it is then removed from the model. Likewise, the variable outside the model that most enhances its discriminatory power is added. The process stops when the performance of the model cannot be enhanced by the addition of more features.

Another method to reduce the dimensionality of the problem is Principal Components Analysis, which is a surface projection method that selects a reduced set of orthogonal (uncorrelated) linear combinations of the variables that explain the greatest part of the variance of the samples (Duda et al., 2000).

In many works, authors compare the efficiency of different classification methods, normally a Bayesian classifier against artificial neural networks. Classifiers based on artificial neural networks are non-statistical and fall outside the scope of this chapter, but are profusely cited in the literature, as will be seen below.

The aim of this chapter is to show applications of Bayesian learning techniques in computer vision in agriculture and the agri-food sector, and to discuss the major challenges that appear in them. As will be seen, computer vision is commonly used for the automatic evaluation of fruit quality, but it is also widely employed in other areas such as in robotic guidance, remote sensing or weed control, all of which require an appropriate classification of pixels for colour image segmentation or for the classification of objects (fruits, weeds, grains, etc.) that are present in the scene.

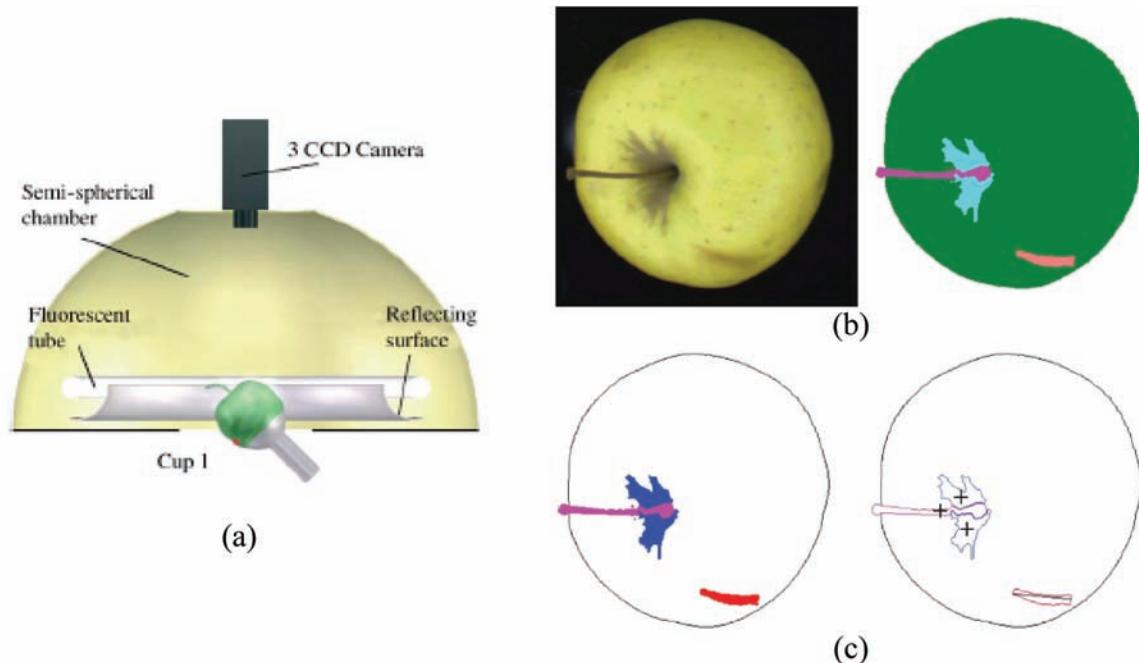
INSPECTION AND CLASSIFICATION OF FRUIT AND VEGETABLES

The aim of inspection and classification of fruit and vegetables is to sort product units into commercial categories of different prices and to reject defective units before they can be marketed. This is one of the most important tasks in the added value chain, since it has a direct influence on consumers' final perception of quality.

The use of machine vision for sorting fruits and vegetables has increased in recent years, with important efforts being carried out to improve the objectivity and the speed of the inspection. Nowadays, several manufacturers around the world produce sorting machines capable of pre-grading fruits by size, colour and weight. Nevertheless, the market constantly requires higher quality products and, consequently, enhancement of machine vision inspection systems is needed to gain an accurate assessment of the properties of the product.

Several problems related with the natural variability of colour, sizes and shapes of biological products make segmentation of images particularly difficult. Fruits and vegetables, as living organisms, evolve after being harvested, which includes sometimes fast, sometimes subtle changes of colour and appearance. Moreover, due to irregular lighting, it is possible to appreciate similar colours in the sound tissues of some pieces and in the defective tissues of others from the same batch, which increases the possibility of confusion of an inspection system based on colour. Furthermore, accuracy in the measurement of size is related to image resolution, and the higher the resolution of the image is, the more processing time is needed, which greatly influences the performance of sorting machines. Bayesian discriminant analysis offers a tool to segment images in a fast simple way, which works well with a reasonable number of classes, allowing on-line, real-time operation of sorting machines when required.

Figure 1. (a) Inspection chamber of the robot SHIVA; (b) Original image of an apple and its segmentation; (c) Detection of defects and stem contours



Ideally, images of fruits and vegetables are divided into a few regions of interest belonging to background, sound surface, defective surface and, in some cases, stem and secondary colours (i.e. variegated apples and pears, some peaches, some aubergines, etc.). The high degree of variability of the pixel colours in these classes reduces the performance of classifiers, because it makes it difficult to describe a regular, connected surface with which to define each class in the colour space. For instance, remember that under the assumption of Gaussian probability distributions, the Bayesian approach produces ellipsoid-like clusters for the patterns of each class, which would be difficult to fit to all colours of sound skin of a variegated apple. For this reason, many authors define more than one colour class to be assigned to each region of interest.

For example, the robot SHIVA (Blasco et al., 2003) was designed to handle, sort and pack fragile fruit (apples, peaches, etc.). Four images of each fruit in different positions were acquired in an inspection chamber (Figure 1) and processed to obtain an average primary and secondary colour, position of the stem, size and presence of defects. A Bayesian image segmentation model was created using the RGB coordinates as independent variables. This involved estimating the probabilities for each combination of RGB values in an image of belonging to any of a number of pre-defined colour classes. Because different covariance matrices were estimated for each class, this was a quadratic discriminant model. Two colour classes were assigned to the sound skin (bright and dark skin), another two to the defective surface, two more to background (light and dark background) and one to the stem.

By joining the pixels with a colour class that was different from the stem and background, the size of a piece of fruit was calculated as the average of the sizes calculated in the four images. By analysing the pixels belonging to each of the regions of interest separately, the algorithm was able to perform an

on-line estimation of the quality of the fruit in terms of the following attributes: size, colour, stem location and detection of external blemishes. A robotic arm moved the piece of fruit into the corresponding packing box. The automated inspection system was tested on-line and the precision and repeatability of the system were found to be similar to those achieved with manual grading.

Since apples are one of the most widely produced fruits in the world, numerous descriptions of image processing applied to the assessment of their quality have appeared in the literature. Many of them use Bayesian approaches. For instance, Tao et al., (1995) use these techniques to classify potatoes and apples. In both cases, a pooled covariance matrix was estimated, which led to LDA. The pooled covariance matrix was obtained from representative samples of each category and the classification was based on hue features. Preliminary data treatment using stepwise discriminant analysis and other techniques helped to reduce the dimensionality of the problem.

Bayesian classification was also successfully used to segment defects on apple skin surfaces (Leemans et al., 1999). In this work, the authors calculated the colour frequency distributions of the healthy tissue and of the defects in order to estimate the probability distribution of colours in each class. The results showed that most defects were adequately segmented, although russet was sometimes confused with some defects. Later, Leemans and Destain (2004) used Quadratic Discriminant Analysis to grade apples. Grades were based on features extracted from defects, calyx and stem portions, which they called blobs. Blobs were characterised by their colour, relative position, shape and textural features. In this case, Principal Component Analysis was used to reduce the number of features needed by the classifier.

Sometimes detection of blemishes is not enough for inspection, since some skin damage may cicatrize while others, like fungal infestations, evolve and contaminate other fruit. Blasco et al., (2007a) proposed a preliminary segmentation using a region-growing algorithm capable of detecting 95% of the external defects in five varieties of oranges and mandarins. In a second step (Blasco et al., 2007b), they used Bayesian discriminant analysis both to identify these defects and to find the best colour space for this purpose. They concluded that the highest success rate was obtained by using the HSI colour space, which was capable of correctly classifying 87% of defects.

Other fruits, such as raisins, table olives and tomatoes, have also been inspected and classified using Bayesian methods in machine vision applications. Okamura et al., (1993) considered texture, shape, size and luminance to separate the raisins into three grades. Then, a Bayes classifier was implemented in which the expected loss in assigning a random sample to each class was computed. The class that resulted in minimum loss was assigned to the corresponding random sample. The computer vision system graded raisins with an accuracy and precision comparable to those of current manual grading methods.

Pydipati et al., (2006) used discriminant analysis to identify citrus diseases based on textural features extracted from the sides of leaves. Previous stepwise analyses were conducted to reduce the dimensionality of the texture feature set. The discriminant functions were a measure of the generalized squared distance between texture variables extracted from an object and the average of these features in each class. It should be remembered that this distance is proportional to the posterior probability of an object's belonging to a class.

LDA or Mahalanobis distances are often used for segmentation and grading purposes. Noordam et al., (2000) built an inspection machine based on image analysis to grade potatoes by colour and shape. The RGB values of the pixels were classified into six different colour classes for segmentation and sorting purposes. However, the system needed different colour models for different potato cultivars. Moreover, it required different shape models for each potato cultivar. Sorting was based on two LDA classifiers, one that used the RGB values for defect detection and the other used the first ten Fourier coefficients for

shape grading. Similarly, Shearer and Payne (1990) used discriminant analysis to classify bell peppers according to colour and presence of damage.

Very often authors compare the performance of Bayesian classifiers against other techniques. Abdul-lah et al., (2006), working on starfruit, used stepwise analysis to select a number of hue-related features. The redundancy and dimensionality of the data were reduced from 65 correlated hue values to 12 uncorrelated principal components. Results indicated that discriminant analysis had a slightly better success rate than the neural network used in their study.

Technological improvements have opened up the possibility of using not only colour but also hyperspectral images (images obtained in several wavelengths). Polder et al., (2002) obtained these images from tomatoes and compared the classification results of linear discriminant models using standard RGB images and hyperspectral images composed of 257 bands. Experimental results showed that the classification error of individual pixels was reduced from 51% to 19% using the hyperspectral images.

Bayesian classifiers have also been applied to X-ray images. Shahin et al., (2002) employed this technique for sorting sweet onions into good or defective classes, based on the presence of internal defects. They performed preliminary stepwise analysis to reduce the dimension of the feature space.

INSPECTION AND CLASSIFICATION OF GRAINS AND SEEDS

Bayesian methods have also been widely used to classify tubers, vegetables, grains or seeds. Majumdar and Jayas (2000) conducted studies aimed at classifying individual kernels of wheat, barley, oats and rye using morphological, colour and textural features. A total of 48 features were employed and, for this reason, a more reduced set was needed. Rankings of all features were determined from correlation coefficients and the average squared canonical correlation. More recently, Choudhary et al., (2008) built linear and quadratic discriminant models assuming normal distribution of wavelet, morphological, colour and textural features of non-touching kernel images.

Zayas et al., (1990) employed computer vision to automatically discriminate whole from broken corn kernels. Multivariate discriminant analysis was used to develop the classification rules to identify whole and broken pieces. They employed basic morphological parameters such as area, perimeter, length, width and convex perimeter together with another eight derived from them. The squared partial correlation coefficient was used as an indicator of the contribution made by each variable to the classification model.

Granitto et al., (2005) proposed a system for the automatic identification of weed seeds from colour and black and white images. Size, shape, colour and texture features were obtained by standard image-processing techniques, and their discriminating power was assessed with standard sequential forward and backward stepwise selection algorithms. They also considered the performance of a naïve Bayes classifier, assuming normal distributions of the class-conditional probabilities in order to reduce the number of features. The work was carried out on a database of 236 different weed species. They compared this simple Bayesian approach with artificial neural networks and concluded that the Bayesian classifier based on an adequately selected set of features had an excellent performance, similar to that of the comparatively more sophisticated neural network approach.

WEED DETECTION AND PLANT IDENTIFICATION

Modern agriculture is oriented towards reducing all kind of inputs (energy, fertilizers, plant protection chemicals, etc.) in order to save costs and reduce its environmental impact. This has led to the development of what is known as Precision Agriculture, aimed at limiting the inputs to the specific areas of the farm where they are required. Precision Agriculture requires the development of sensors that detect the local status of the different parts of the field and the technology for variable rate application of inputs.

Weeding is a very time-consuming task. Farmers dedicate great amounts of time and labour to this task. Moreover, modern societies are concerned about the impact of agricultural practices on the environment and for this reason efforts have been made to reduce the use of chemical herbicides around the world. Precision weeding relies on sensors that detect weeds and distinguish them from the crop. Computer vision therefore plays an important role in the development of such techniques.

In this context, robotic weeding was envisaged as a means to put Precision Agriculture into practice. European Project Patchwork was aimed at building an autonomous robot capable of detecting weeds and eliminating them by physical (non-chemical) means. Particularly, a vision system was implemented to locate the weeds in order to guide a robotic arm that produced an electrical discharge to kill the weeds, without damaging the crop.

Blasco et al., (2002) used the Bayesian approach to segment images by tagging the pixels as belonging to crop, weed or soil categories. As previously seen, the RGB coordinates were used to classify the pixels. Shadows, irregularities in the natural illumination (due to the relative position of the sun with respect to the imaged scene, the weather conditions of that particular instant, etc.) plus the inherent variability of weeds, crop and soil give rise to changes in the appreciation of their colour from one image to another. Again, different colour categories are used to define each of the selected regions of interest (weed, crop, soil). For example, several colour regions were assigned to weeds and to crop. Furthermore, the class *soil* was built from two groups of colours. However, it has to be taken into account that this solution offers a major drawback because most classifiers reduce their success rate as the number of classes increases (Duda et al., 2000).

The Bayesian decision model was constructed assuming an equal *a priori* probability of pixels' being included in the pre-defined colour categories. Pixels belonging to these colour categories, once joined, constituted the regions of interest, i.e. weed, crop and soil. In the first step, images were scanned and each pixel was automatically assigned to the class *plant* (weed or crop) or *soil*, depending on its RGB coordinates. This method was fast enough to achieve real-time time requirements, but did not separate the weeds from the crop. Pixels were correctly classified in 96% of cases.

Thus, further processing was needed to distinguish between weeds and crop. In the above-mentioned segmented image, continuous clusters of pixels of the same class were considered to be an object and the area, the perimeter and the centroid of each of these objects were calculated. Objects smaller than a pre-established threshold were considered to be noise, and were then removed and assigned to the neighbouring object. Objects larger than another preset threshold were considered to be crop, and remaining objects were considered to be weeds, with a success rate of 84% in detecting weeds and 99% in detecting plants.

Identification of weeds is a complementary task. In this regard, many works can be found in the literature. For instance, Burks et al., (2000) used discriminant analysis to identify five weed species grown in pots. They used hue and saturation features to build their Bayesian classifier instead of the typical RGB coordinates.

Similarly, Neto et al., (2006) used discriminant analysis with Elliptic Fourier (EF) features to inspect the shapes of leaves in order to identify four plant species. Stepwise discriminant analysis was performed on EF coefficients to reduce the feature set.

Other authors have searched for objective methods for selecting filter combinations to facilitate the detection of weeds located within rows. Piron et al., (2007) acquired images of real carrot crops under artificial lighting over a period of 19 days and with different filters, starting one week after crop emergence, which is the period when the presence of weeds has the most negative effect on yield. The best combination of interference filters was selected by computing the highest classification rate obtained by Quadratic Discriminant Analysis.

As in many other applications, some authors have compared the efficiency of Bayesian and Artificial neural network classifiers. Vrindts et al., (2002) used models based on RGB ratios of canopy reflectance and reported that the best accuracy was obtained by using discriminant analysis.

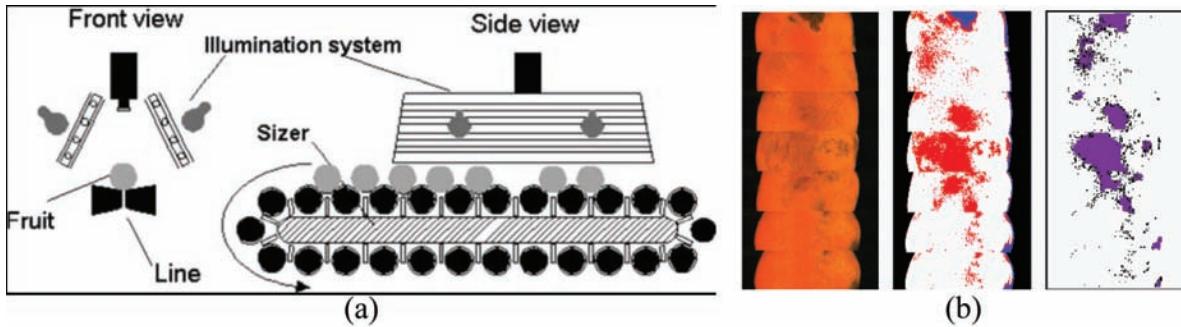
INSECT SEXING IN A BIO-FACTORY

The Mediterranean fruit fly (*Ceratitis capitata*, Wiedemann) is one of the most important pests that affect many fruits (citrus, peaches, figs, etc.). One biological method to control this pest is based on the mass delivery of sterile males that copulate with females in the wild without producing offspring, thus reducing the population of the pest.

Male pupae are artificially reared in bio-factories, where they are sterilised. Subsequently, these pupae are sent to the distribution areas, where they develop into adult flies that are released by planes. Progenitors in the bio-factory (the so-called colony) have artificially induced genetic characteristics, which facilitates the manipulation of their descendants. One important genetic characteristic that they have is that the puparia of females are white and the puparia of males are brown, which makes pupae identification a simple matter. However, descendants may experience spontaneous mutations that do not express this desired characteristic (these are called recombinant individuals). It is essential for the bio-factory to prevent the introduction of recombinant individuals into the colony when it is renewed in the normal production process. Detecting recombinants is an extremely slow and painstaking task, since sexing is based on identifying small, distinctive features under a magnifying glass. These are due to sexual dimorphism, and may include the presence of two bristles with enlarged diamond-shaped tips (spatulate setae) in males, or an ovipositor in females. When the sex of an emerged fly does not correspond with the colour of its puparium (male emerged from a whitish puparium or female emerged from a dark or brown puparium), the individual is recombinant and must be eliminated. Operators work in very short stints to prevent the decisions they make from being affected by fatigue. It is therefore very important to automate the process.

Blasco et al., (2008a) built a machine and used computer vision techniques to capture and analyse images of living insects in order to find these elements of sexual dimorphism and consequently determine the sex of an individual automatically. The image analysis algorithm extracted the contour of the abdomen and used the polar and the arc-length versus cumulative turning angle graphs. The Fast Fourier Transform (FFT) of these graphs was calculated to obtain the harmonics that were to be used to classify different type of contours. The first 10 harmonics obtained for each graph (20 variables) were used as independent variables to build a Bayesian classifier. Assuming the same *a priori* probability for all the classes, classification functions were calculated for each class to determine which sex group (*male*,

Figure 2. (a) Scheme of the real-time inspection system for citrus fruits; (b) Original colour image of the fruit surface (left), its segmented image (centre) and the defects that were detected (right)



(female or undetermined) each particular individual was more likely to belong to. The classification functions were utilised to classify 5000 images of flies that were different from those used in the training set. Tests had a 100% success rate in identifying male flies with an error rate of 0.6% for female flies, thus proving to be accurate enough to be installed in a bio-factory.

IMPLEMENTATION OF BAYESIAN SEGMENTATION IN REAL-TIME APPLICATIONS

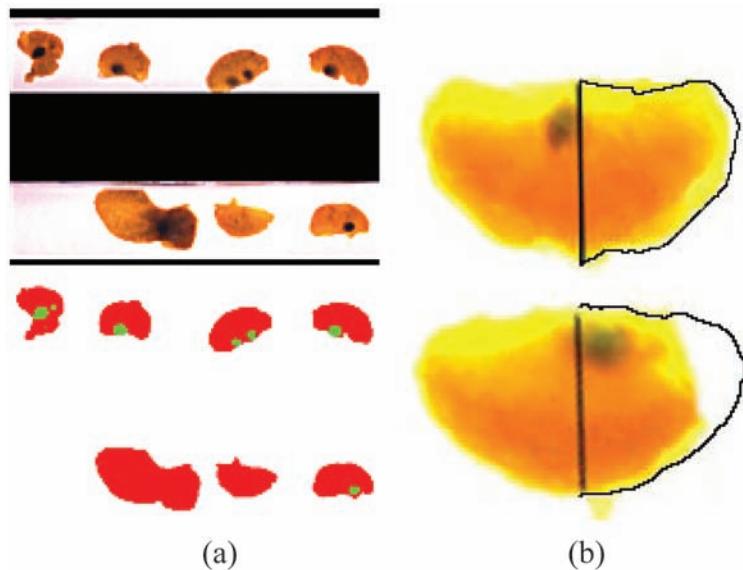
The aim of many computer vision systems is to replace human operators so as to achieve a more objective and faster automatic operation. Automatic inspection of fruit and vegetables requires very fast machine vision systems if they are to reach commercially acceptable performances. A typical citrus sorting machine, for instance, must inspect about 15 or 20 pieces of fruit per second. Moreover, a single image of a fruit is not enough to examine its whole surface and, normally, in order to ensure a complete inspection, several images of the fruit are acquired from different views.

In order to reduce processing time, most industrial machines today take low resolution images and implement fast algorithms to discriminate only between the sound skin of the fruit, the background and the largest, more highly contrasted skin defects. Small-sized blemishes produced by insects, fungus, physiological disorders or small impacts are not detected. Moreover, the reduced information and the lack of time, make it very difficult to distinguish between the different types of defects in order to achieve an adequate identification.

The development of new, higher resolution cameras and faster processors allows more information to be acquired from the scenes as well as the implementation of more sophisticated algorithms, not only for detecting skin blemishes but also for identifying their cause.

The capacities of real-time image processing have been boosted by the introduction of specific hardware implementations, such as the use of specific integrated circuits (ASIC), digital signal processors (DSP) or field programmable gate arrays (FPGA). Aleixos et al., (2002) designed a real-time inspection system for citrus based on a quadratic Bayesian classifier. They designed the software to run in parallel in two DSP, thus decreasing the inspection time enormously. In this work, they also showed a multi-spectral camera designed to acquire the same scene in near-infrared (NIR) and the three conventional

Figure 3. (a) Original image above and its segmentation below; (b) Symmetry of an entire segment above and a broken segment below



RGB colour bands. The grey levels of pixels in these four bands could be used as independent variables to build the classifier, which was programmed to distinguish between background, sound peel, defects and stem. Fruits were classified attending to their size, colour and defects.

The whole surface of the fruit was inspected because the pieces were rotated while travelling below the camera. The system used two types of images at the same time. One type contained a picture of the whole scene in NIR, which was used to estimate the size and shape of the fruit. The other type of image was generated while the fruit rotated, by considering only the central strip of the fruit. Up to 15 shots of the fruit in different positions were used to construct this image, which contained RGB and NIR information and was used for global colour estimation and for the detection of skin defects.

This approach allowed the authors to build a parallel algorithm by dividing the inspection tasks between the two DSP, running in parallel in a master/slave architecture. The master processor calculated the geometrical and morphological features of the fruit in the NIR pictures, and the slave processor estimated the fruit colour and the skin defects using the NIR and RGB bands in the image formed by the addition of the strips (Figure 2).

Real-time applications of Bayesian classifiers can take advantage of the use of Look Up tables (LUT) to accelerate the image processing software. An example of this is presented for the classification of table olives in a real-time inspection application by Diaz et al., (2000). In this work the ultimate objective was to classify the olives into four categories. In a previous off-line process, an expert manually selected different regions of interest on a training set of images that represented the colour variability of the product. The operator assigned each selected region to one of the predefined classes (bright background, dark background, bright olive, dark olive, bright defect, dark defect). The RGB values of each selected pixel were stored, together with its class. Once training was finished, the covariance matrix and the means were calculated in order to generate the Bayesian classifier. In a second step, the LUT was generated by computing the predicted class of all the possible RGB combinations. This table associated

Figure 4. (a) Detection and feature extraction application; (b) Results of grading into four categories



all the possible RGB values with one of the predefined classes, thus avoiding the need to calculate the classification score for each class for each pixel during on-line operation, which dramatically increased the processing speed. This implementation was used on a machine for the automatic inspection of olives that needed 150 ms per image, each of which had a size of 400×320 mm and contained approximately 66 olives.

In some particular cases, Bayesian segmentation algorithms can be replaced by simpler techniques, like thresholding. This was highlighted by Blasco et al., (2008b) in the development of a computer vision-based machine for the on-line inspection and classification of mandarin segments. Backlighting was also used to produce a high contrast between segments and the background (Figure 3a).

After segmentation, the system extracted morphological features from the objects in order to automatically identify pieces of skin and undesired material; it also separated entire segments from broken ones, as well as discriminating between those that were slightly broken and those with larger breakages. Morphological features such as length, area, perimeter, symmetry (Figure 3b), compactness, shape factor or elongation were employed to build a non-linear Bayesian classifier that was able to correctly detect 95% of sound segments.

Blasco et al., (2008c) also presented an innovative system for grading pomegranate arils to be marketed as fresh packed fruit. In order to obtain real-time specifications and to achieve real-time operation, two image segmentation methods (Figure 4a) were tested: one used a threshold on the R/G ratio and the other was based on Bayesian Linear Discriminant Analysis in the RGB space. LDA offered an average success rate of 92% on a validation set, while success reached 89% (Figure 4b) with the thresholding method. In this application, subsequent classification of objects was based on colour and geometric features using a Bayesian algorithm.

Separation of the arils into categories was performed by air ejectors. The movement of the conveyor belts was tracked by an optical encoder attached to the shaft of the carrier roller and connected to the serial port of the computer. This work reports the results of industrial testing in commercial fruit facilities.

FUTURE TRENDS

In the field of the agriculture and the agro-industry, new machine vision applications based on the use of cameras sensitive to infrared have been derived to complement the traditional color information obtained from conventional cameras. The possibility of acquiring images at wavelengths invisible to the human naked eye is becoming simpler and easier thanks to the reduction of the price of this technology. Bayes-

ian based segmentation, feature selection and object classification will be a partial or total solution for image understanding and their adequate exploitation.

In a similar sense hyperspectral image systems are increasingly being used in our field. Techniques that have been used for remote sensing are currently investigated for the inspection of horticultural products, robotics, yield forecast, etc. However, Hyperspectral images produce a significant increase in the amount of data to be processed, while the needs of high processing speed remain and even increase. Methods and techniques that are easy to implement, robust and able to manage a large number of variables simultaneously are still required and Bayesian techniques can fulfill such necessities.

CONCLUSION

Agricultural commodities have a great variability in terms of sizes, colors, textures, shapes and presence of defects. Fruits and vegetables, as living organisms, evolve after being harvested, which includes changes of colour and appearance. This variability makes necessary to work with a large number of variables that must be taken into account to produce a decision.

In such complex, automated inspection systems there is a need to work with high-resolution images, compromising the processing time, which greatly influences the performance of sorting machines. Typically, inspection systems operate at speeds that require the implementation of classification algorithms capable of processing images in milliseconds, and this can be achieved by the use of classifiers based in Bayesian inference theory as shown in this chapter.

Bayesian theory provides as a tool for developing robust, supervised algorithms to classify pixels and objects in a very fast way and for this reason it is still commonly used in many applications within the agriculture and the agro-industry. In this chapter we have shown illustrative examples like on-line inspection of fruits and vegetables, robotics, quality assessment of grain, or even sexing insects for new pest control systems.

REFERENCES

- Abdullah, M. Z., Saleh, J. M., Fathinul-Syahir, A. S., & Mohd-Azemi, B. M. N. (2006). Discrimination and classification of fresh-cut starfruits (*Averrhoa carambola L.*) using automated machine vision system. *Journal of Food Engineering*, 76, 506–523. doi:10.1016/j.jfoodeng.2005.05.053
- Aleixos, N., Blasco, J., Navarrón, F., & Moltó, E. (2002). Multispectral inspection of citrus in real-time using machine vision and digital signal processors. *Computers and Electronics in Agriculture*, 33, 121–137. doi:10.1016/S0168-1699(02)00002-9
- Blasco, J., Aleixos, N., Gómez-Sanchis, J., & Moltó, E. (2007b). Citrus sorting identification of the most common defects using multispectral computer vision. *Journal of Food Engineering*, 83, 384–393. doi:10.1016/j.jfoodeng.2007.03.027
- Blasco, J., Aleixos, N., & Moltó, E. (2003). Machine vision system for automatic quality grading of fruit. *Biosystems Engineering*, 85(4), 415–423. doi:10.1016/S1537-5110(03)00088-6

Blasco, J., Aleixos, N., & Moltó, E. (2007a). Computer vision detection of peel defects in citrus by means of a region oriented segmentation algorithm. *Journal of Food Engineering*, 81, 535–543. doi:10.1016/j.jfoodeng.2006.12.007

Blasco, J., Aleixos, N., Roger, J. M., Rabatel, G., & Moltó, E. (2002). Robotic weed control using machine vision. *Biosystems Engineering*, 83(2), 149–157. doi:10.1006/bioe.2002.0109

Blasco, J., Cubero, S., Gómez-Sanchís, J., Mira, P., & Moltó, E. (2008c). Development of a machine for the automatic sorting of pomegranate (*Punica granatum*) arils based on computer vision. *Journal of Food Engineering*, 90(1), 27–34. doi:10.1016/j.jfoodeng.2008.05.035

Blasco, J., Cubero, S., Gómez-Sanchis, J., & Moltó, E. (2008b). Citrus sorting identification of the most common defects using multispectral computer vision. *Lecture Notes in Computer Science*, 5112, 1071–1080. doi:10.1007/978-3-540-69812-8_107

Blasco, J., Gómez-Sanchis, J., Gutierrez, A., Chueca, P., Argiles, R., & Moltó, E. (2008a). Automatic sex detection of individuals of Ceratitis Capitata by means of computer vision in a biofactory. *Pest Management Science*, 65(1), 99–104. doi:10.1002/ps.1652

Burks, T. F., Shearer, S. A., & Payne, F. A. (2000a). Classification of weed species using color texture features and discriminant analysis. *Transactions of the American Society of Agricultural Engineers*, 43(2), 441–448.

Choudhary, R., Paliwal, J., & Jayas, D. S. (2008). Classification of cereal grains using wavelet, morphological, colour, and textural features of non-touching kernel images. *Biosystems Engineering*, 99, 330–337. doi:10.1016/j.biosystemseng.2007.11.013

Chtioui, Y., Panigrahi, S., & Backer, L. F. (1999). Rough sets theory as a pattern classification tool for quality assessment of edible beans. *Transactions of the American Society of Agricultural Engineers*, 42(4), 1145–1152.

Diaz, R., Faus, G., Blasco, M., Blasco, J., & Moltó, E. (2000). The application of fast algorithm for the classification of olives by machine vision. *Food Research International*, 33, 305–309. doi:10.1016/S0963-9969(00)00041-7

Domenico, S., & Gary, W. (1994). Machine vision and neural nets in food processing and packaging—natural way combinations. In *Food processing automation III—Proceedings of the FPAC conference* (pp. 11). Orlando, FL: ASAE.

Duda, R. O., Hart, P. E., & Stork, D. G. (2000). *Pattern classification* (2nd ed.). New York: Wiley-Interscience.

Granitto, P. M., Verdes, P. F., & Ceccatto, H. A. (2005). Large-scale investigation of weed seed identification by machine vision. *Computers and Electronics in Agriculture*, 47, 15–24. doi:10.1016/j.compag.2004.10.003

James, M. (1985). *Classification algorithms*. London: Ed. Collins.

Leemans, V., & Destain, M. F. (2004). A real-time grading method of apples based on features extracted from defects. *Journal of Food Engineering*, 61(1), 83–89. doi:10.1016/S0260-8774(03)00189-4

- Leemans, V., Magein, H., & Destain, M. F. (1999). Defect segmentation on 'Jonagold' apples using colour vision and a Bayesian classification method. *Computers and Electronics in Agriculture*, 23, 43–53. doi:10.1016/S0168-1699(99)00006-X
- Majumdar, S., & Jayas, D. S. (2000). Classification of cereal grains using machine vision: IV. Combined morphology, color, and texture models. *Transactions of the American Society of Agricultural Engineers*, 43(6), 1689–1694.
- Marchant, J. A., & Onyango, C. M. (2003). Comparison of a Bayesian classifier with a multilayer feed-forward neural network using the example of plant/weed/soil discrimination. *Computers and Electronics in Agriculture*, 39, 3–22. doi:10.1016/S0168-1699(02)00223-5
- Mitchell, R. S., Sherlock, R. A., & Smith, L. A. (1996). An investigation into the use of machine learning for determining oestrus in cows. *Computers and Electronics in Agriculture*, 15(3), 195–213. doi:10.1016/0168-1699(96)00016-6
- Neto, J. C., Meyer, G. E., Jones, D. D., & Samal, A. K. (2006). Plant species identification using Elliptic Fourier leaf shape analysis. *Computers and Electronics in Agriculture*, 50, 121–134. doi:10.1016/j.compag.2005.09.004
- Noordam, J. C., Otten, G. W., Timmermans, A. J. M., & Zwol, B. V. (2000). High speed potato grading and quality inspection based on a color vision system. In K. W. Tobin Jr. (Ed.), *Machine vision applications in industrial inspection VIII, Proceedings of SPIE* (Vol. 3966) (pp. 206-220).
- Okamura, N. K., Delwiche, M. J., & Thompson, J. F. (1993). Raisin grading by machine vision. *Transactions of the American Society of Agricultural Engineers*, 36(2), 485–492.
- Onyango, C. M., Marchant, J. A., & Zwiggelaar, R. (1997). Modeling uncertainty in agriculture image analysis. *Computers and Electronics in Agriculture*, 17, 295–305. doi:10.1016/S0168-1699(97)01322-7
- Piron, A., Leemans, V., Kleynen, O., Lebeau, F., & Destain, M. F. (2008). Selection of the most efficient wavelength bands for discriminating weeds from crop. *Computers and Electronics in Agriculture*, 62(2), 141–148. doi:10.1016/j.compag.2007.12.007
- Polder, G., Van der Heijden, G. W. A. M., & Young, I. T. (2002). Spectral image analysis for measuring ripeness of tomatoes. *Transactions of the American Society of Agricultural Engineers*, 45(4), 1155–1161.
- Pydipati, R., Burks, T. F., & Lee, W. S. (2006). Identification of citrus disease using color texture features and discriminant analysis. *Computers and Electronics in Agriculture*, 52, 49–59. doi:10.1016/j.compag.2006.01.004
- Ripley, B. D. (1996). *Pattern recognition and neural networks*. Cambridge, UK: Cambridge University Press.
- Shahin, M. A., Tollner, E. W., Gitaitis, R. D., Sumner, D. R., & Maw, B. W. (2002). Classification of sweet onions based on internal defects using image processing and neural network techniques. *Transactions of the American Society of Agricultural Engineers*, 45(5), 1613–1618.

Shearer, S. A., & Payne, F. A. (1990). Color and defect sorting of bell peppers using machine vision. *Transactions of the American Society of Agricultural Engineers*, 33(6), 2045–2050.

Tao, Y., Heinemann, P. H., Varghese, Z., Morrow, C. T., & Sommer, H. J. III. (1995). Machine vision for color inspection of potatoes and apples. *Transactions of the American Society of Agricultural Engineers*, 38(5), 1555–1561.

Zayas, I., Converse, H., & Steele, J. (1990). Discrimination of whole from broken corn kernels with image analysis. *Transactions of the American Society of Agricultural Engineers*, 33(5), 1642–1646.

KEY TERMS AND DEFINITIONS

Bayesian Techniques: Set of statistical techniques which involve the Bayes theorem in order to get a probabilistic approach of the problem.

Machine Vision: Machine vision constitute an approach to analyze image content based on a global set of image descriptors, such as color features, combined with image transforms, in order to extract knowledge.

Stem Recognition: Machine vision procedure used in agriculture applications of fruit sorting. This process must be used in order to avoid misclassification errors in fruits defects detection.

Fourier Descriptors: In this chapter, it refers to harmonics of Fast Fourier Transform of signal constituted by the distances from mass center to perimeter points of the object.

Precision Agriculture: Nowadays, the agriculture techniques are aimed to reducing the costs and limiting the environmental impact.

Ceratitis Capitata, Wiedeman: the Mediterranean fruit fly or medfly, is a species of fruit fly capable of wreaking extensive damage to a wide range of fruit crops.

NIR (Near InfraRed): Electromagnetic spectral range between 700 nm and 1100 nm.

LUT (Look Up Table): A data structure, usually an array or associative array, often used to replace a runtime computation with a simpler array indexing operation.

Image Segmentation: Is the process of partitioning a digital image composed by pixels into multiple regions of interest.

Hyperspectral Imaging: Hyperspectral data is a set of contiguous bands acquired across the electromagnetic spectrum usually by one sensor.

Multispectral Imaging: Multispectral data is a set of optimally chosen spectral bands that are typically not contiguous and can be collected from multiple sensors.

Real Time Operation: A process that is computed fitting time constraints to meet a required deadline.

Chapter 24

Software Cost Estimation using Soft Computing Approaches

K. Vinaykumar

Institute for Development and Research in Banking Technology (IDRBT), India

V. Ravi

Institute for Development and Research in Banking Technology (IDRBT), India

Mahil Carr

Institute for Development and Research in Banking Technology (IDRBT), India

ABSTRACT

Software development has become an essential investment for many organizations. Software engineering practitioners have become more and more concerned about accurately predicting the cost of software products to be developed. Accurate estimates are desired but no model has proved to be successful at effectively and consistently predicting software development cost. This chapter investigates the use of the soft computing approaches in predicting the software development effort. Various statistical and intelligent techniques are employed to estimate software development effort. Further, based on the above-mentioned techniques, ensemble models are developed to forecast software development effort. Two types of ensemble models viz., linear (average) and nonlinear are designed and tested on COCOMO'81 dataset. Based on the experiments performed on the COCOMO'81 data, it was observed that the nonlinear ensemble using radial basis function network as arbitrator outperformed all the other ensembles and also the constituent statistical and intelligent techniques. The authors conclude that using soft computing models they can accurately estimate software development effort.

INTRODUCTION

Software development has become an important activity for many modern organizations (Pressman, 1997). In fact the quality, cost, and timeliness of developed software are often crucial determinants of an organization's success. There are significant financial and strategic implications for development projects in terms of activity scheduling and cost estimation. Software cost estimation is one of the most

DOI: 10.4018/978-1-60566-766-9.ch024

critical tasks in managing software projects. Development costs tend to increase with project complexity and hence accurate cost estimates are highly desired during the early stages of development (Wittig and Finnie, 1997). A major problem of the software cost estimation is first obtaining an accurate size estimate of the software to be developed (Kitchenham et al., 2003). An important objective of the software engineering community is to develop useful models that constructively explain the software development life cycle and accurately estimate the cost of software development.

In order to effectively develop software in an increasingly competitive and complex environment many organizations use software metrics as a part of their project management process. The field concerned with managing software development projects using empirical models is referred to as software project management (Fenton and Pleeger, 1997). Software metrics are aspects of software development (either of the software product itself, or of the development process producing that product) that can be measured. These measurements can be used as variables in models for predicting or estimating some aspects of the development process or product that are of interest.

Estimating development effort and schedule, can include activities such as assessing and predicting system quality, measuring system performance, estimating user satisfaction and in fact any modeling task involving measurable attributes of interest within the software development sphere (Gray, 1999). However, the most researched area has been effort estimation as it carries the greatest promise of benefit for project management. Such models are generally developed using a set of measures that describe the software development process, product and resources-such as developer experience, system size, complexity, and the characteristics of the development environment respectively. The output of the model is usually some measure of effort in terms of person hours (months or years).

There are many models and tools used in software cost estimation that provide invaluable information regarding efforts and expenditure to the management to bid for a project (Kitchenham et al., 2003). The most commonly used methods for predicting software development effort have been based on linear-least-squares regression such as COCOMO (Fenton and Pleeger, 1997; Pressman, 1997). As such, the models have been extremely susceptible to local variations in data points (Miyazaki et al., 1994). Additionally, the models have failed to deal with implicit nonlinearities and interactions between the characteristics of the project and effort (Gray, 1999).

In recent years, a number of alternative modeling techniques have been proposed. Existing data sets have their performance examined with some success including those in Gray and MacDonell (1997). Alternative models include artificial neural networks, analogy based reasoning, regression trees and rule induction models. Gray and MacDonell (1997) applied fuzzy logic to software metric models for development effort estimation. They outlined the use of fuzzy logic for defining software metrics as linguistic variables and for modeling processes. They made comparison of results obtained from an elementary fuzzy inference system with other techniques such as linear regression and neural network techniques and found that it outperformed them well. Gray (1999) presented several different predictive model-building techniques such as robust statistical procedures, various forms of neural network models, fuzzy logic, case-based reasoning and regression trees. He also described a simulation-based study on the performance of these empirical modeling techniques using size and effort software metric dataset and observed that M-estimation regression outperformed all other parametric and non-parametric techniques. Xu and Khoshgoftaar (2004) presented an innovative fuzzy identification software cost estimation modeling technique, which is an advanced fuzzy logic technique that integrates fuzzy clustering, space projection, fuzzy inference and defuzzification. Based upon their case study on the COCOMO'81 database it was observed that the fuzzy identification model provided significantly better cost estimations

Table 1. Summary of neural network studies

Study	Learning algorithm	Dataset	Number of projects	Predicting	Results (MMRE)
Venkatachalam (1993)	Back-propagation	COCOMO	63	Development effort	Promising
Wittig & Finnie (1997)	Back-propagation	Desharnais / ASMA	81 136	Effort	17%
Jorgensen (1995)	Back-propagation	Jorgenson	109	Maintenance effort	100%
Serluca (1995)	Back-propagation	Mermaid-2	28	Development effort	76%
Samson et al. (1993)	Back-propagation	COCOMO	63	Development effort	428%
Srinivasan & Fisher (1995)	Back-propagation	Kermel & CO-COMO	78	Development effort	70%
Hughes (1996)	Back-propagation	Hughes	33	Development effort	55%

than the three COCOMO models, i.e., Basic, Intermediate and Detailed. Many researchers have applied the neural networks approach to estimate software development effort (Hughes, 1996; Jorgensen, 1995; Samson et al., 1993; Schofield, 1998; Seluca, 1995; Venkatachalam, 1993; Srinivasan and Fisher, 1995 and Vinaykumar et al., 2008). Most of their investigations have focused more attention on the accuracy of the other cost estimation techniques. Table 1 shows a summary of artificial neural networks effort prediction studies. Idri et al., (2002) have also done research on estimating software cost using the neural networks approach and fuzzy if-then rules on the COCOMO'81 dataset. Most recently, Vinaykumar et al., (2008) employed wavelet neural network (WNN) to forecast the software development effort. They used two variants of WNN with Morlet function and Gaussian function as transfer function and also proposed threshold acceptance training algorithm for wavelet neural network (TAWNN). The effectiveness of the WNN variants is compared with multilayer perceptron (MLP), radial basis function network (RBFN), multiple linear regression (MLR), dynamic evolving neuro-fuzzy inference system (DENFIS) and support vector machine (SVM) in terms of mean magnitude relative error (MMRE) obtained on the Canadian financial (CF) dataset and the IBM data processing services (IBMDPS) dataset. Based on the experiments conducted, they observed that the WNN-Morlet for CF dataset and WNN-Gaussian for IBMDPS outperformed all the other techniques. Also, TAWNN outperformed all other techniques except WNN.

In this chapter, we are concerned with cost estimation models that are based on artificial neural networks, statistical techniques, neuro fuzzy techniques and decision tree techniques. The present chapter deals with application of soft computing approaches i.e. using stand alone techniques and design of ensemble models for forecasting the software development effort using intelligent techniques from a surrogate measure of software size (i.e., source lines of code).

The rest of the chapter is organized as follows: In Section 2, we briefly describe the techniques applied to software cost estimation. In Section 3, the ensembles developed here are presented. Section 4 presents our experimental methodology and a discussion of the results. Finally, Section 5 concludes the chapter.

SOFT COMPUTING TECHNIQUES FOR SOFTWARE COST ESTIMATION

Basically, soft computing is an association of computing methodologies that includes as its principal members fuzzy logic (FL), neuro computing (NC), evolutionary computing (EC) and probabilistic computing (PC). An essential aspect of soft computing is that its constituent methodologies are, for the most part, complementary and symbiotic rather than competitive and exclusive. Therefore, soft computing systems involve seamless integration of the intelligent technologies viz. fuzzy logic (FL), neuro computing (NC), evolutionary computing (EC) and probabilistic computing (PC) (Zadeh, 1998). The interest and confidence in techniques such as SVM, DENFIS, PSN, TANN, RBF and GRNN has grown enormously during the past 5 to 10 years. Statistical techniques are no longer preferred in view of their low accuracy. Further, the BPNN is also very much exploited. Other NN architectures deserve much wider application in this field. However, decision trees such as CART are not employed as much as they deserve.

The constituents of the soft computing models developed here are Neural Network Techniques (Multilayer Perceptron (MLP), Threshold-Accepting-based Neural Network (TANN), Radial Basis Function Network (RBFN), Pi-Sigma Network (PSN) and General Regression Neural Network (GRNN)), statistical approaches (Multiple Linear Regression (MLR), Multivariate Adaptive Regression Splines (MARS) and Classification and Regression Trees (CART)), Neuro-Fuzzy techniques (Dynamic Evolving Neuro-Fuzzy Inference System (DENFIS)) and Support Vector Machine (SVM).

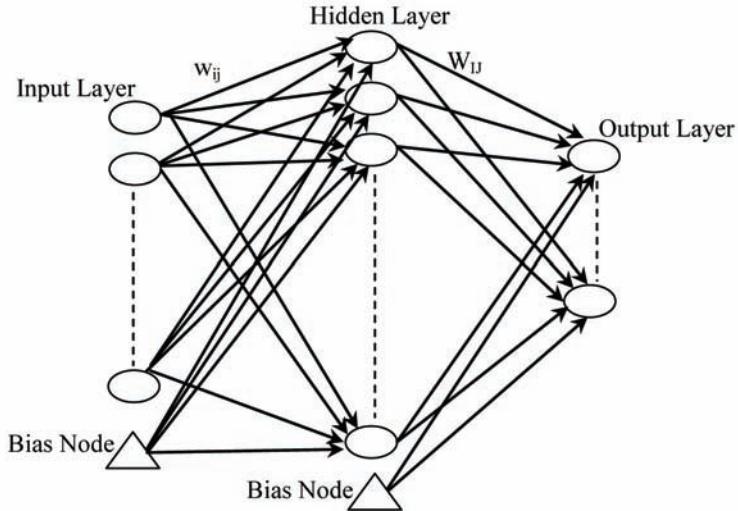
Neural Network Techniques

An artificial neural network (ANN) is typically composed of several layers of many computing elements called nodes. Each node receives an input signal from other nodes or external inputs and then after processing the signals locally through a transfer function, it outputs a transformed signal to other nodes or final result. ANNs are characterized by the network architecture, that is, the number of layers, the number of nodes in each layer and how the nodes are connected. Many different models of neural networks have been proposed. They may be grouped into two major categories. First, feed-forward networks are networks where no loops occur in the path. Second, feedback networks have recursive loops. The feed forward multi-layer perceptron with back-propagation learning algorithm are the most commonly used in the cost estimation field. Idri et al., (2002) applied multi layer perceptron with back-propagation learning algorithm.

Multilayer Perceptron (MLP)

In a popular form of ANN called the multi-layer perceptron (MLP), all nodes and layers are arranged in a feed forward manner (depicted in Figure 1). The first or the lowest layer is called the input layer where external information is received. The last or the highest layer is called the output layer where the network produces the model solution. In between, there are one or more hidden layers which are critical for ANNs to identify the complex patterns in the data. Acyclic arcs from a lower layer to a higher layer connect all nodes in adjacent layers. Three-layer MLP is a commonly used ANN structure for two-group classification problems like the bankruptcy prediction. The parameters (arc weights) of a neural network model need to be estimated before the network can be used for prediction purposes.

Figure 1. Architecture of Multi-layer Perceptron



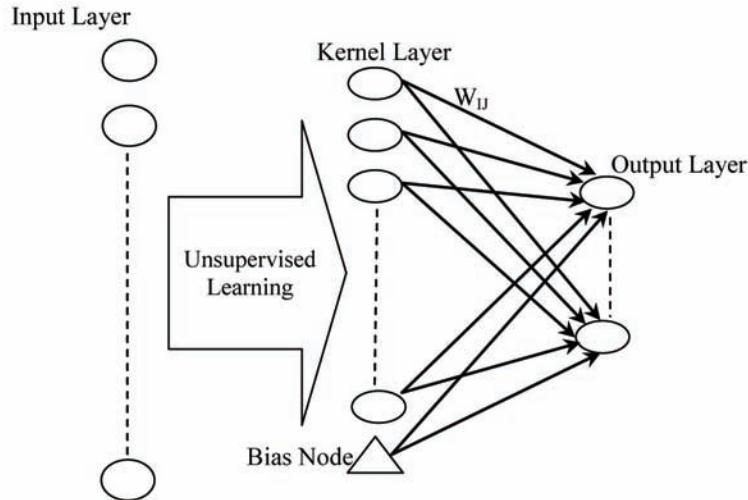
The process of determining these weights is called training. The training phase is a critical part in the use of neural networks.

For classification problems, the network training is a supervised one in that the desired or target response of the network for each input pattern is always known a priori. During the training process, patterns or examples are presented to the input layer of a network. The activation values of the input nodes are weighted and accumulated at each node in the hidden layer. The weighted sum is transferred by an appropriate transfer function into the node's activation value. It then becomes an input into the nodes in the output layer. Finally an output value is obtained to match the desired value. The aim of training is to minimize the differences between the ANN output values and the known target values for all training patterns. From this perspective, network training is an unconstrained nonlinear minimization problem. The most popular algorithm for training is the well-known backpropagation (Williams et al., 1986), which is basically a gradient steepest descent method with a constant step size. Due to problems of slow convergence and inefficiency with the steepest descent method, many variations of backpropagation have been introduced for training neural networks. For a prediction problem, only one output node is needed. The output values from the neural network (the predicted outputs) are used for prediction. MLP is coded by the second author in C.

RBFN

RBFN (Moody and Darken, 1989), another member of the feed-forward neural networks, has both unsupervised and supervised phases. In the unsupervised phase, input data are clustered and cluster details are sent to hidden neurons, where radial basis functions of the inputs are computed by making use of the center and the standard deviation of the clusters (depicted in Figure 2). The radial basis functions are similar to kernel functions in kernel regression. The activation function or the kernel function can use a variety of functions, though Gaussian radial basis functions are the most commonly used. The learning between hidden layer and output layer is of supervised learning type where ordinary least squares technique is used. As a consequence, the weights of the connections between the kernel layer (also called

Figure 2. Architecture of Radial Basis Function Neural Network



hidden layer) and the output layer are determined. Thus, it comprises of a hybrid of unsupervised and supervised learning. RBFN can solve many complex prediction problems with quite satisfying performance. Like MLP, RBFN is also a universal approximator. However, training an RBFN is very fast. Depending on data set, MLP and RBFN are capable of outperforming each other. RBF as implemented in MATLAB (www.mathworks.com) is used in this chapter.

Threshold-Acceptance-based Neural Network (TANN)

Threshold accepting (TA), originally proposed by Dueck and Scheur (1990), is a faster variant of the original Simulated Annealing algorithm wherein the acceptance of a new move or solution is determined by a deterministic criterion rather than a probabilistic one. The predominant aspect of TA is that it accepts any new solution, which is not much worse than the current one. The crux of the TA-based training algorithm (Ravi et al., 2005; Ravi and Zimmermann, 2001; Ravi and Zimmermann, 2003) for the feed forward neural networks is that the ‘forward pass’ of the back propagation algorithm is not disturbed and retained ‘as it is’. But, in the backward pass, which essentially updates all the weights, TA is used instead of the steepest descent algorithm used in back propagation. In this context, the set of weights of the neural network (both input to hidden and hidden to output nodes) becomes the vector of decision variables. TANN is coded by the second author in C.

Pi-Sigma Network (PSN)

The Pi-Sigma network (PSN) was originally proposed by Ghosh and Shin (1991). It is a feed-forward network with a single hidden layer, where the number of hidden units (also called as ‘summing units’) represents the order of the network, which can be varied as required. In the output layer there are product units whose output is a function of the product of the individual summing units’ output. In each iteration of the algorithm, until the convergence criterion is met, one of the summing units will be selected at random and the corresponding weights of the links connected to that node are updated according to a rule similar to the delta rule. PSN is coded by the first author in C.

General Regression Neural Network (GRNN)

GRNN introduced by Specht (1991) can be thought of as a normalized Radial Basis Function (RBF) network in which there is a hidden unit centered at every training case. These RBF units are called “kernels” and are usually probability density functions such as the Gaussian. The hidden-to-output weights are just the target values, so the output is simply a weighted average of the target values of training cases close to the given input case. The only weights that need to be learned are the widths of the RBF units. These widths (often a single width is used) are called “smoothing parameters” or “bandwidths” and are usually chosen by cross-validation or by more esoteric methods that are not well known in the neural net literature; gradient descent is not used. GRNN is a universal approximator for smooth functions, so it should be able to solve any smooth function-approximation problem given enough data. The main drawback of GRNN is that, it suffers badly from the curse of dimensionality. GRNN cannot ignore irrelevant inputs without major modifications to the basic algorithm. GRNN as implemented in MATLAB (www.mathworks.com) was used in this chapter.

Statistical Techniques

Statistics is a mathematical science pertaining to the collection, analysis, interpretation or explanation, and presentation of data. It is applicable to a wide variety of fields from the physical and social sciences to the humanities. In statistics, regression analysis is used to model relationships between variables and determine the magnitude of those relationships. The models can be used to make predictions.

Multiple Linear Regression (MLR)

The general purpose of multiple regressions is to learn more about the relationship between several independent (or predictor) variables and a dependent (or criterion) variable. MLR (implemented using SPSS – www.spss.com) performs a least squares fit on multivariate data. The method takes a data set with several input variables and one output variable (continuous values) and finds a formula that approximates linearly the data samples. The error between the approximated and the desired output values is used as an objective function. This is also called “a data fitting method”. After the regression formula is generated from the data, it can be used as a prediction model for new input vectors (Rawlings, 1988).

Multivariate Adaptive Regression Splines (MARS)

Friedman (1991) introduced Multivariate Adaptive Regression Splines (MARS). MARS automates the development and deployment of accurate and easy to understand regression models. MARS (<http://www.salford-systems.com>) has been used to build business intelligence solutions for problems such as predicting credit card holder balances, insurance claim losses, customer catalog orders, and cell phone usage. It excels at finding optimal variable transformations and interactions, the complex data structure that often hides in high-dimensional data. In doing so, this new approach to regression modeling effectively uncovers important data patterns and relationships that are difficult, if not impossible, for other methods to reveal.

Classification and Regression Tree (CART)

Decision trees form an integral part of ‘machine learning’ an important sub-discipline of artificial intelligence. Almost all the decision tree algorithms are used for solving classification problems. However, algorithms like CART (classification and regression trees) solve regression problems also. Decision tree algorithms induce a binary tree on a given training set, resulting in a set of ‘if-then’ rules. These rules can be used to solve the classification or regression problem. CART (<http://www.salford-systems.com>) is a robust, easy-to-use decision tree tool that automatically sifts large, complex databases, searching for and isolating significant patterns and relationships. CART uses a recursive partitioning, a combination of exhaustive searches and intensive testing techniques to identify useful tree structures in the data. This discovered knowledge is then used to generate a decision tree resulting in reliable, easy-to-grasp predictive models in the form of ‘if-then’ rules. CART is powerful because it can deal with incomplete data, multiple types of features (floats, enumerated sets) both in input features and predicted features, and the trees it produces often contain rules, which are humanly readable. Decision trees contain a binary question (yes/no answer) about some feature at each node in the tree. The leaves of the tree contain the best prediction based on the training data. Decision lists are a reduced form of this where one answer to each question leads directly to a leaf node. A tree’s leaf node may be a single member of some class, a probability density function (over some discrete class) or a predicted mean value for a continuous feature or a Gaussian (mean and standard deviation for a continuous value). The key elements of a CART analysis are a set of rules for: (i) Splitting each node in a tree (ii) Deciding when a tree is complete; and (iii) Assigning each terminal node to a class outcome (or predicted value for regression). CART has been used to solve problems such as profiling customers, targeting direct mailings and company bankruptcy prediction.

Neuro-Fuzzy Techniques

Neuro-fuzzy refers to hybrids of artificial neural networks and fuzzy logic. Neuro-fuzzy hybridization results in a hybrid intelligent system that synergizes these two techniques by combining the human-like reasoning style of fuzzy systems with the learning and connectionist structure of neural networks. Neuro-fuzzy hybridization is widely termed as Fuzzy Neural Network (FNN) or Neuro-Fuzzy System (NFS) in the literature.

Dynamic Evolving Neuro-Fuzzy Inference System (DENFIS)

Before we describe DENFIS, fuzzy ‘if-then’ rule are briefly defined. Fuzzy if-then rules are the rules whose antecedents, consequences or both are fuzzy rather than crisp. The most widely used interpretation considers a fuzzy rule “if x is A then y is B ” as a fuzzy point $A \times B$ and a collection of fuzzy rules “if x is A_i then y is B_i ” $i = 1, \dots, n$ as a fuzzy graph providing a rough description of the relation between x and y (e.g., Zadeh, 1992, 1994). The benefit of having fuzzy antecedents is to provide a basis for an interpolation mechanism. From this perspective, a fuzzy rule is defined by means of a conjunction (for defining a fuzzy Cartesian product $A \times B$) rather than in terms of a multiple-valued logic implication (Dubois and Prade, 1996).

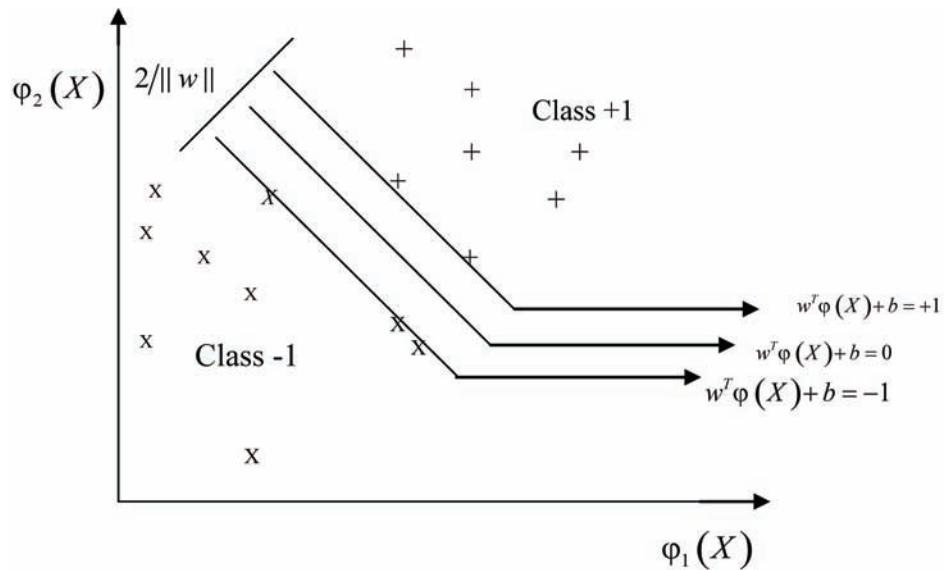
Kasabov and Song (2002) introduced DENFIS, which is a neuro-fuzzy system. It evolves through incremental, hybrid (supervised/unsupervised), learning, and accommodate new input data, including

new features, new classes, etc., through local element tuning. New fuzzy rules are created and updated during the operation of the system. At each time moment, the output of DENFIS is calculated through a fuzzy inference system based on -most activated fuzzy rules, which are dynamically chosen from a fuzzy rule set. A set of fuzzy rules can be inserted into DENFIS before or during its learning process. Fuzzy rules can also be extracted during or after the learning process. DENFIS as implemented in NeuCom (www.theneucom.com) is used in this chapter for conducting experiments.

Support Vector Machines (SVMs)

The SVM (depicted in Figure 3) is a powerful learning algorithm based on recent advances in statistical learning theory (Vapnik, 1998). SVMs are learning systems that use a hypothesis space of linear functions in a high dimensional space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory (Cristianini and Shawe-Taylor, 2000). SVMs have recently become one of the popular tools for machine learning and data mining and can perform both classification and regression. SVM uses a linear model to implement nonlinear class boundaries by mapping input vectors nonlinearly into a high-dimensional feature space using kernels. The training examples that are closest to the maximum margin hyperplane are called support vectors. All other training examples are irrelevant for defining the binary class boundaries. The support vectors are then used to construct an optimal linear separating hyperplane (in case of pattern recognition) or a linear regression function (in case of regression) in this feature space. The support vectors are conventionally determined by solving a quadratic programming (QP) problem. SVMs have the following advantages: (i) they are able to generalize well even if trained with a small number of examples (ii) they do not assume prior knowledge of the probability distribution of the underlying data set. SVM is simple enough to be analyzed mathematically. In fact, SVM may serve as a sound alternative combining the advantages of conventional statistical methods that are more theory-driven and easy to analyze and machine learning

Figure 3. Illustration of SVM optimization of the margin in the feature space



methods that are more data-driven, distribution-free and robust. (<http://www.svms.org/introduction.html>). Recently, SVM has been used in financial applications such as credit rating, time series prediction and insurance claim fraud detection. The tool used in this chapter is LibSVM, which can be downloaded from <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.

A METHODOLOGY FOR EVALUATING SOFTWARE COST ESTIMATION APPROACHES

This section presents the methodology followed in this work, in applying soft computing techniques for estimating the software cost estimation on COCOMO'81 dataset. In software cost estimation, we have to estimate the effort depending upon cost drivers and a measure of software size i.e. Kilo Delivered Source Instructions (KDSI). We employ the following criteria to assess and compare the performance of cost estimation models.

Evaluation Criteria

A common criterion for the evaluation of cost estimation model is the magnitude of relative error (MRE), which is defined as:

$$MRE = \left| \left(\frac{\text{Effort}_{\text{actual}} - \text{Effort}_{\text{estimated}}}{\text{Effort}_{\text{actual}}} \right) \right|$$

The MRE values are calculated for each project, while MMRE aggregates the n projects. For N multiple projects, we can also use mean magnitude of relative error (MMRE):

$$MMRE = \frac{1}{N} \sum_{x=1}^N MRE_x$$

Generally, the acceptable target value for MMRE is 25%. This indicates that on the average, the error of the established estimation models would be less than 25%. Since MMRE is well-known evaluation criteria, they are adopted as estimation performance indicators in the present chapter. Another widely used criterion is the Pred (L) which represents the percentage of MRE that is less than or equal to the value L among all projects. This measure is often used in literature and is the proportion of the projects for a given level of accuracy. The definition of Pred (L) is shown as:

$$\text{Pred}(L) = \frac{k}{n}$$

Where n is the total number of observations and k is the number of observations whose MRE is less than or equal to L. A common value for L is 0.25, which is also used in the present study. The Pred (0.25) represents the percentage of projects whose MRE is less than or equal to 25%. The Pred (0.25)

value identifies the effort estimates that are generally accurate while the MMRE is fairly conservative with a bias against overestimates.

We conducted two types of experiments. First the above-discussed statistical and intelligent techniques were applied on COCOMO'81 dataset. The dataset used here is COCOMO'81 dataset from (Boehm, 1981). COCOMO dataset contains 63 projects, and each project contains 15 cost multipliers, KDSI (Kilo Delivered Source Instructions) using which we estimated effort of the software. The cost driver attributes are grouped into four categories: software product attributes, computer attributes, personnel attributes, and project attributes. They are *Product Attributes (RELY: Required Software Reliability, DATA: Database Size and CPLX: Product Complexity), Computer Attributes (TIME: Execution Time Constraint, STOR: Main Storage Constraint, VIRT: Virtual Machine Volatility and TURN: Computer Turnaround Time), Personnel Attributes (ACAP: Analyst Capability, AEXP: Applications Experience, PCAP: Programmer Capability, VEXP: Virtual Machine Experience and LEXP: Programming Language Experience) and Project Attributes (MODP: Modern Programming Practices, TOOL: Use of Software Tool, SCED: Required Development Schedule)*. But all the 15 cost drivers are not equally important in calculating the software cost (Idri et al., 2002). We observe that Gray and MacDonell (1997) considered only 12-cost drivers those are *RELY, DATA, TIME, STOR, VIRT, TURN, ACAP, AEXP, PCAP, VEXP, LEXP and SCED*. In this chapter we followed a similar approach and considered 12 cost drivers. Architecture of our model is shown in Figure 4.

On examining the dataset we observed that effort (MM) and software size (KDSI) have a very high variability in scale. Hence, we need to preprocess the original COCOMO'81 database. Because of high variability in scale of MM and KDSI columns in COCOMO'81 dataset, we have applied logarithmic transformation on them, and then applied the above-discussed statistical and intelligent techniques. We call the resulting transformed dataset as modified COCOMO dataset. The modified dataset is split into training and test sets in the ratio of 80%: 20%. All the models are trained with training data and results on test data are presented in Table 2.

Nine-Fold Cross Validation

If one constructs an effort estimation model using a particular data set and then computes the accuracy of the model using the same data sets, the accuracy evaluation may be too optimistic. The estimation error may be artificially low and does not reflect the performance of the model on another unseen data set. A cross-validation approach gives a more realistic accuracy assessment. It involves dividing the whole data set into multiple training and test sets. The estimation results in the training stages present the estimation accuracies from the model construction data sets.

The test stage evaluates the estimation accuracies of the models using the other unseen data sets. In the training stage, a cross-validation approach calculates the accuracy for each training data set, and then combines the accuracies across all training result. In the test stage, it calculates the accuracy for each test set, and then combines the accuracies across all test sets for the test result. We also performed cross validation i.e. nine-fold cross validation using only 6 techniques. We selected Multiple linear Regression, Multilayer Perceptron, Radial Basis Functions, Dynamic Evolving Neuro Fuzzy Inference System, Support Vector Regression and Threshold Acceptance Neural Network. The remaining four techniques viz., General Regression Neural Network, Pi-sigma Network, Classification and Regression Tree (CART) and Multivariate Adaptive Regression Splines (MARS) are not considered because they yielded worse results than earlier studies in the hold-out method as can be seen from Table 2. By considering ten-folds

Figure 4. Architecture for Software development effort

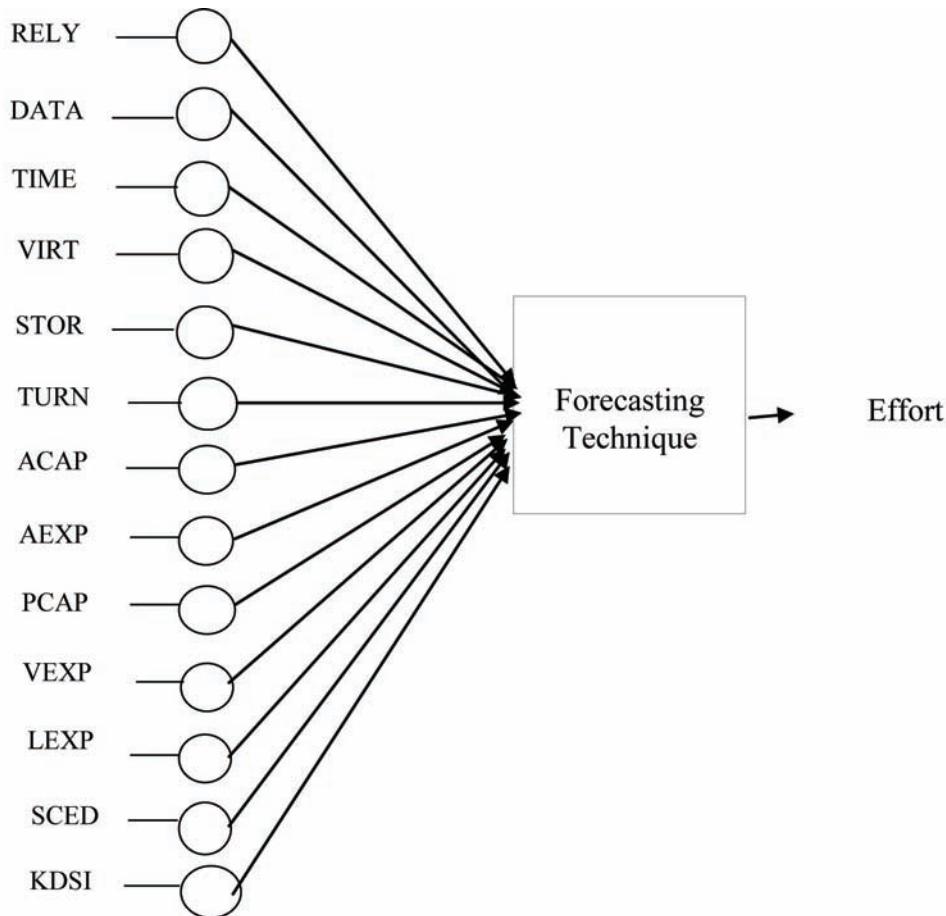


Table 2. MMRE values of techniques in hold-out method

Technique	MMRE Value
DENFIS	0.347803318
MLR	0.358138498
MLP	0.295070
RBF	0.320085
GRNN	0.572609
PSN	0.486697
CART	0.55906
MARS	0.652858
SVM	0.325689542
TANN	0.341155

3 projects are left out resulting in odd division. Hence, for the purpose of proper division of data we considered nine-fold cross validation.

COCOMO dataset contains 63 projects. We divided the data into nine-folds, each consists of 7 projects. In each experiment, one of the nine folds will be considered as test set and the remaining eight folds as training set. We applied nine-fold cross validation for all above explained techniques. Table 3 shows the summary of results of nine-fold cross validation.

Ensemble Systems

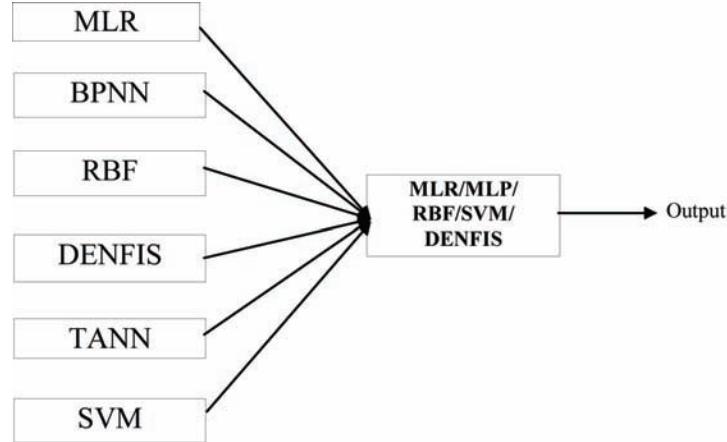
In addition to applying various statistical and intelligent techniques, we also designed linear and nonlinear ensemble forecasting systems. The idea behind ensemble systems is to exploit each constituent model's unique features to capture different patterns that exist in the dataset. Both theoretical and empirical work indicates that ensembling can be an effective and efficient way to improve accuracies. Bates and Granger (1969) in their seminal work showed that a linear combination of different techniques would give a smaller error variance than any of the individual techniques working in stand-alone mode. Since then, many researchers worked on ensembling or combined forecasts. Makridakis et al., (1982) reported that combining several single models has become common practice in improving forecasting accuracy. Then, Pelikan et al., (1992) proposed combining several feed-forward neural networks to improve time series forecasting accuracy. More literature is found in the review given by Clemen (1989). Some of the ensemble techniques for prediction problems with continuous dependent variable include linear ensemble e.g., simple average (Benediktsson et al., 1997), weighted average (Perrone and Cooper, 1993) and stacked regression (Breiman, 1996) and nonlinear ensemble e.g., neural-network-based nonlinear ensemble (Yu et al., 2005).

In Hansen et al., (1992) it was reported that the generalization ability of a neural network system could be significantly improved by using an ensemble of a number of neural networks. The purpose is to achieve improved overall accuracy on the production data. In general, for classification problems, an ensemble system combines individual classification decisions in some way, typically by a majority voting to classify new examples. The basic idea is to train a set of models (experts) and allow them to vote. In majority voting scheme, all the individual models are given equal importance. Another way of

Table 3. MMRE results of nine-fold cross validation in hold-out method

Fold/ Technique	MLR	DENFIS	RBF	MLP	TANN	SVM
FOLD-1	0.35166	0.33	0.174549	0.524997	0.605259	0.291802
FOLD-2	0.36508	0.38	0.137133	0.426265	0.489785	0.367995
FOLD-3	1.22	1.20	0.6106640	0.892248	2.04	0.556662
FOLD-4	0.216034	0.2746	0.201223	0.380494	0.581581	0.226968
FOLD-5	0.44376	0.39	0.418631	0.455422	1.211438	0.324843
FOLD-6	0.705	0.70	0.386047	0.619703	1.277253	1.044685
FOLD-7	0.45	0.22725	0.197203	0.513837	0.357395	0.432631
FOLD-8	0.38	0.308604	0.131494	0.224594	0.421493	0.374256
FOLD-9	0.2170	0.378037	0.240841	0.359973	0.336013	0.264824
AVERAGE	0.48317	0.465388	0.277532	0.488615	0.813357	0.431629

Figure 5. The non-linear ensemble



combining the models is via weighted voting, wherein the individual models are treated as unequally important. Attaching weights to the prediction given by the individual models and then combining them achieve this.

Six different types of ensembles are constructed here based upon (i) non-linear ensemble (uses one neural network technique or neuro fuzzy technique or regression technique) (ii) linear ensemble (Average), which are depicted in Figure 5 & Figure 6 respectively. In the linear ensemble based on average, for each observation, the output values of the individual components are taken as the input to the ensemble and the average of these values is output by the ensemble. This is the simplest kind of ensemble one can imagine. In neural network based non-linear ensemble, here, there are no assumptions that are made about the input that is given to the ensemble. The output values of the individual techniques are fed into an arbitrator, 5 arbitrators viz., MLP, RBF, MLR, DENFIS and SVM are used as shown in Figure 5. Those 5 techniques that yielded better results in 9-fold cross validation which is why we considered them for ensemble design.

Figure 6. The linear ensemble

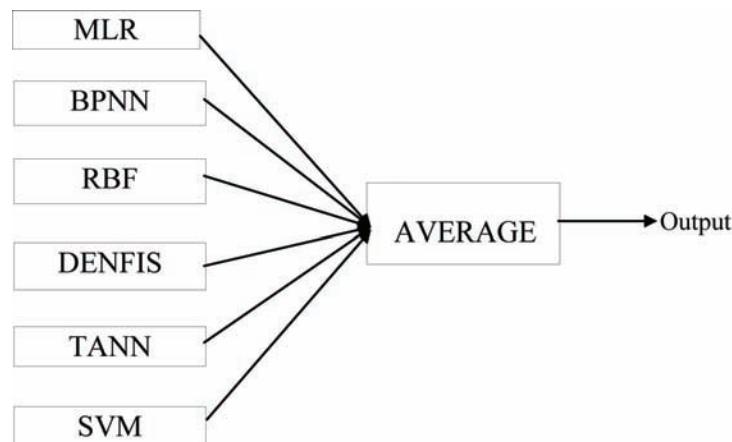


Table 4. MMRE of all ensembles with different arbitrators

Fold / Arbitrator	Average	MLR	MLP	RBF	DENFIS	SVM
FOLD-1	0.297046	0.169493	0.230813	0.156249	0.167483	0.170881
FOLD-2	0.343396	0.136482	0.18585	0.13649	0.130356	0.142646
FOLD-3	1.208214	0.613585	0.616763	0.576262	0.613109	0.614488
FOLD-4	0.272022	0.20507	0.20466	0.162961	0.196983	0.2071
FOLD-5	0.46131	0.420436	0.656529	0.243922	0.42556	0.407572
FOLD-6	0.662455	0.385691	0.597962	0.212459	0.344421	0.38463
FOLD-7	0.169007	0.201039	0.209322	0.1902	0.205462	0.194204
FOLD-8	0.282515	0.131952	0.19	0.083987	0.129484	0.129126
FOLD-9	0.280256	0.240352	0.228211	0.183558	0.264824	0.221469
AVERAGE	0.441802	0.278233	0.346679	0.216232	0.275298	0.27468

RESULTS AND DISCUSSION

Interesting observations can be drawn from Tables 2, 3 and 4. MLP outperforms all the other techniques in the holdout method, although MLR, DENFIS, RBF, TANN and SVM performed well. However, other techniques such as CART, MARS, PSN and GRNN have not performed well because the MMRE values produced by them are higher than that of Idri et al., (2002). After MLP, the techniques of RBF, SVM, TANN, DENFIS and MLR produced good results in that order in terms of MMRE value. After applying nine-fold cross validation on modified COCOMO'81 data, RBF outperformed all the other techniques. Other than TANN, all other techniques viz., MLR, DENFIS, MLP and SVM performed consistently well. TANN performed well in holdout testing but compared to nine-fold cross validation. The results obtained by all techniques other than RBF in nine-fold cross validation are not satisfactory compared to their holdout method. Hence, several ensembles are designed and developed using the above techniques. We performed nine-fold cross validation on modified COCOMO'81 dataset using linear and nonlinear ensembles. The MMRE values obtained from linear and nonlinear ensembles are presented in Table 4.

The nonlinear ensemble with Radial-Basis Function (RBF) yielded better cost estimation than the COCOMO'81 model. For example, RBF based ensemble predicted the effort more accurately for every project resulting in an MMRE of 21%. Nonlinear ensembles with DENFIS and SVR also yielded better cost estimation than the COCOMO'81 model. Except linear ensemble and MLP based ensemble remaining all other ensembles obtained better cost estimation than COCOMO'81 model. The nonlinear ensembles based on DENFIS, SVR and MLR have given consistently same accuracy. The techniques of RBF, DENFIS, SVM and MLR produced good results in that order in terms of MMRE value. From Tables 2-4 we can say that MMRE is improved from 32% to 21% for RBF. RBF has not performed well in the hold-out method but improved in the nine-fold cross validation and in ensemble mode also. The MMRE value for DENFIS is improved from 0.34 to 0.27 in ensemble mode. Not only RBF and DENFIS all other techniques improved their MMRE value in ensemble mode when compared with holdout method.

Table 5 presents the average values of Pred (0.25) and Pred (0.10) in 9-fold cross-validation for all the ensemble models. Pred (0.25) denotes the percentage of projects whose MRE are equal to or less than 0.25, and Pred (0.10) denotes the percentage of projects whose MRE are equal to or less than

Table 5. Average values of Pred (0.25) and Pred (0.10) in 9-fold cross-validation over all ensembles

Arbitrator used in ensemble model	Pred (0.25)	Pred (0.10)
DENFIS	0.65	0.348111
MLR	0.602667	0.348667
MLP	0.571	0.219778
RBF	0.729778	0.428111
AVERAGE	0.330556	0.171778
SVM	0.555	0.315889

0.10. It is clear that in terms of Pred (0.25) and Pred (0.10) also, ensemble model with RBF arbitrator outperformed all other techniques.

CONCLUSION AND FUTURE RESEARCH

It is a well-known fact that software project management teams can greatly benefit from knowing the estimated cost of their software projects. The benefits can have greater impact if accurate cost estimations are deduced during the early stages of the project life cycle. Estimating the process and performance behavior of a software project, early in the software life cycle, is very challenging and often very difficult to model. This issue is further elevated by the fact that important and useful recorded information pertaining to the software cost are often vague, imprecise, and in some cases even linguistic. Traditionally used software cost estimation techniques are not capable of incorporating such vague and imprecise information in their cost estimation models. This incapability prevents the extraction and use of important information that could very well improve a model's project cost estimation. The COCOMO cost estimation model has been commonly used in obtaining software cost and effort estimations. The model incorporates data collected from several projects, and uses this gathered information for its costs and effort estimations. However it may not provide accurate project cost estimations as the software size and complexity increases. In this chapter, one linear ensemble and six nonlinear ensembles are developed and tested to forecast software development effort. Various statistical and intelligent techniques constitute the ensembles. They are Neural Network Techniques (Multilayer Perceptron, Threshold-accepting-based neural network (TANN), Radial Basis Function Network (RBFN), Pi-Sigma Network, General Regression Neural Network), Statistical approaches (Multiple Linear Regression (MLR), Multivariate Adaptive Regression Splines (MARS), Classification and Regression Trees (CART)), Neuro-Fuzzy techniques (Dynamic Evolving Neuro-Fuzzy Inference System (DENFIS)) and Support Vector Machine (SVM). Based on the numerical experiments conducted by us on the COCOMO'81 dataset, we infer that the nonlinear ensemble using Radial Basis Function neural network outperformed all the other ensembles and also the constituent statistical and intelligent techniques. Finally, we conclude that the present chapter is a significant milestone in providing accurate forecasts for the software development effort in a firm. Future research can replicate and confirm this estimation technique with fresh datasets.

REFERENCES

- Bates, J. M., & Granger, C. W. J. (1969). The combination of forecasts. *Operational Research Quarterly*, 20, 451–468. doi:10.1057/jors.1969.103
- Benediktsson, J. A., Sveinsson, J. R., Ersoy, O. K., & Swain, P. H. (1997). Parallel consensual neural networks. *IEEE Transactions on Neural Networks*, 8, 54–64. doi:10.1109/72.554191
- Boehm, B. W. (1981). *Software engineering economics*. Englewood Cliffs, NJ: Prentice-Hall.
- Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24, 123–140.
- CART. (2009). *Salford Systems Inc*. Retrieved from <http://www.salford-systems.com>
- Clemen, R. (1989). Combining forecasts: A review & annotated bibliography with discussion. *International Journal of Forecasting*, 5, 559–608. doi:10.1016/0169-2070(89)90012-5
- Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge, UK: Cambridge University Press.
- Dubois, D., & Prade, H. (1996). What are fuzzy rules and how to use them. *Fuzzy Sets and Systems*, 84, 169–185. doi:10.1016/0165-0114(96)00066-8
- Dueck, G., & Scheuer, T. (1990). Threshold accepting: A general-purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, 90(1), 161–175. doi:10.1016/0021-9991(90)90201-B
- Fenton, N. E., & Pleeger, S. L. (1997). *Software metrics: A rigorous & practical approach* (2nd ed.). Boston, MA: PWS.
- Friedman, J. H. (1991). Multivariate adaptive regression splines (with discussion). *Annals of Statistics*, 19, 1–141. doi:10.1214/aos/1176347963
- Ghosh, J., & Shin, Y. (1991). The pi-sigma networks: An efficient higher-order neural network for pattern classification and function approximation. In *Proceedings of the International Joint Conference on Neural Networks*, Seattle, Washington (pp. 13-18).
- Gray, A., & MacDonell, S. (1997). Applications of fuzzy logic to software metric models for development effort estimation. In *Proceedings of the 1997 Annual meeting of the North American Fuzzy Information Proceeding Society- NAFIPS'97* (pp. 394-399). Washington, DC: IEEE.
- Gray, A. R. (1999). A simulation-based comparison of empirical modeling techniques for software metric models of development effort. In *Proceedings of the International Conference on Neural Information Processing* (pp. 526-531). Washington, DC: IEEE.
- Hansen, J., McDonald, J., & Slice, J. (1992). Artificial intelligence and generalized qualitative response models: An empirical test on two audit decision-making domains . *Decision Sciences*, 23, 708–723. doi:10.1111/j.1540-5915.1992.tb00413.x
- Hughes, R. T. (1996). *An evaluation of machine learning techniques for software effort estimation*. University of Brighton.

Idri, A., Khosgoftaar, T. M., & Abran, A. (2002). Can neural networks be easily interpreted in software cost estimation? In *Proceedings of the 2002 World Congress on Computational Intelligence*, Honolulu, Hawaii (pp. 12-17).

Jorgensen, M. (1995). Experience with accuracy of software maintenance task effort prediction models. *IEEE Transactions on Software Engineering*, 21(8), 674–681. doi:10.1109/32.403791

Kasabov, N. K., & Song, Q. (2002). DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction. *IEEE transactions on Fuzzy Systems*, 10(2), 144–154. doi:10.1109/91.995117

Kitchenham, B., Pickard, L. M., Linkman, S., & Jones, P. W. (2003). Modeling software bidding risks. *IEEE Transactions on Software Engineering*, 29(6), 542–554. doi:10.1109/TSE.2003.1205181

Makridakis, S., Anderson, A., Carbone, R., Fildes, R., Hibdon, M., & Lewandowski, R. (1982). The accuracy of extrapolation (time series) methods: Results of a forecasting competition. *Journal of Forecasting*, 1, 111–153. doi:10.1002/for.3980010202

Miyazaki, Y., Terakado, M., Ozaki, K., & Nozaki, N. (1994). Robust regression for developing software estimation models. *Journal of Systems and Software*, 27(1), 3–16. doi:10.1016/0164-1212(94)90110-4

Moody, J., & Darken, C. J. (1989). Fast learning in networks of locally tuned processing units. *Neural Computation*, 1(2), 281–294. doi:10.1162/neco.1989.1.2.281

Pelikan, E., De Groot, C., & Wurtz, D. (1992). Power consumption in West-Bohemia: Improved forecasts decorrelating connectionist networks. *Neural Network World*, 2(6), 701–712.

Perrone, M. P., & Cooper, L. N. (1993). When networks disagree: Ensemble methods for hybrid neural networks. In R. J. Mammone (Ed.), *Neural networks for speech and image processing* (pp. 126-142). Boca Raton, FL: Chapman Hall.

Pressman, R. S. (1997). *Software engineering: A practitioner's approach* (4th ed.). New York: McGraw-Hill.

Ravi, V., Murty, B. S. N., & Dutt, N. V. K. (2005). Forecasting the properties of carboxylic acids by a threshold accepting-trained neural network. *Indian Chemical Engineer*, 47(3), 147–151.

Ravi, V., & Zimmermann, H. J. (2001). A neural network and fuzzy rule base hybrid for pattern classification. *Soft Computing*, 5(2), 152–159. doi:10.1007/s005000000071

Ravi, V., & Zimmermann, H. J. (2003). Optimization of neural networks via threshold accepting: A comparison with back propagation algorithm. In *Proceedings of the Conference on Neuro-Computing and Evolving Intelligence*, Auckland, New Zealand.

Rawlings, J. O. (1988). *Applied regression analysis: A research tool*. CA: Wadsworth Inc.

Samson, B., Ellison, D., & Dugard, P. (1993). Software cost estimation using an Albus perceptron. In *Proceedings of the 8th International COCOMO Estimation meeting*, Pittsburgh, USA.

Schofield, C. (1998). *Non-algorithmic effort estimation techniques* (Tech. Rep. TR98-01).

- Seluca, C. (1995). *An investigation into software effort estimation using a back-propagation neural network*. Unpublished master's thesis, Bournemouth University, UK.
- Specht, D. F. (1991). A general regression neural network. *IEEE Transactions on Neural Networks*, 2(6), 568–576. doi:10.1109/72.97934
- Srinivasan, K., & Fisher, D. (1995). Machine learning approaches to estimating software development effort. *IEEE Transactions on Software Engineering*, 21(2), 126–136. doi:10.1109/32.345828
- Vapnik, V. N. (1998). *Statistical learning theory*. New York: John Wiley.
- Venkatachalam, A. R. (1993). Software cost estimation using artificial neural networks. In *Proceedings of the International Joint Conference on Neural networks*, Nagoya (pp. 987-990). Washington DC: IEEE.
- Vinaykumar, K., Ravi, V., Carr, M., & Raj Kiran, N. (2008). Software development cost estimation using wavelet neural networks. *Journal of Systems and Software*, 81(11), 1853–1867. doi:10.1016/j.jss.2007.12.793
- Williams, R. J., Rumelhart, D. E., & Hinton, G. E. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition* (pp. 318-362). Cambridge, MA: The MIT Press.
- Wittig, G., & Finnie, G. (1997). Estimating software development effort with connectionist models. *Information and Software Technology*, 39, 469–476. doi:10.1016/S0950-5849(97)00004-9
- Xu, Z., & Khoshgoftaar, T. M. (2004). Identification of fuzzy models cost estimation . *Fuzzy Sets and Systems*, 145, 141–163. doi:10.1016/j.fss.2003.10.008
- Yu, L., Wang, S. Y., & Lai, K. K. (2005). A novel non-linear ensemble-forecasting model incorporating GLAR and ANN for foreign exchange rates. *Computers & Operations Research*, 32(10), 2523–2541. doi:10.1016/j.cor.2004.06.024
- Zadeh, L. A. (1992). The calculus of fuzzy if-then rules. *AI Expert*, 7(3), 23–27.
- Zadeh, L. A. (1994). Soft computing and fuzzy logic. *IEEE Software*, 11(6), 48–56. doi:10.1109/52.329401
- Zadeh, L. A. (1998). Some reflections on soft computing, granular computing and their roles in the conception, design and utilization of information/intelligent systems . *Soft Computing*, 2(1), 23–25. doi:10.1007/s005000050030

KEY TERMS AND DEFINITIONS

Classification and Regression Trees (CART): CART (classification and regression trees) solves classification and regression problems as well. Decision tree algorithms induce a binary tree on a given training set: resulting in a set of ‘if-then’ rules

Dynamic Evolving Neuro-Fuzzy Inference System (DENFIS): It evolves through incremental: hybrid (supervised/unsupervised), learning, and accommodates new input data, including new features, new classes, etc., through local element tuning. New fuzzy rules are created and updated during the operation of the system

Ensemble Forecasting Method: The idea behind ensemble systems is to exploit each constituent model's unique features to capture different patterns that exist in the dataset.:

Multilayer Perceptron (MLP): MLP is typically composed of several layers of many computing elements called nodes. Each node receives an input signal from other nodes or external inputs and then after processing the signals locally through a transfer function: it outputs a transformed signal to other nodes or final result

Radial Basis Function Network (RBFN): RBFN is another member of the feed-forward neural networks and has both unsupervised and supervised phases. In the unsupervised phase: input data are clustered and cluster details are sent to hidden neurons, where radial basis functions of the inputs are computed by making use of the center and the standard deviation of the clusters

Software Cost Estimation: Software development has become an essential investment for many organizations. Software engineering practitioners have become more and more concerned about accurately predicting the cost of software products to be developed.:

Support Vector Machine (SVM): The SVM is a powerful learning algorithm based on recent advances in statistical learning theory. SVMs are learning systems that use a hypothesis space of linear functions in a high dimensional space: trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory

Chapter 25

Counting the Hidden Defects in Software Documents

Frank Padberg
Saarland University, Germany

ABSTRACT

The author uses neural networks to estimate how many defects are hidden in a software document. Input for the models are metrics that get collected when effecting a standard quality assurance technique on the document, a software inspection. For inspections, the empirical data sets typically are small. The author identifies two key ingredients for a successful application of neural networks to small data sets: Adapting the size, complexity, and input dimension of the networks to the amount of information available for training; and using Bayesian techniques instead of cross-validation for determining model parameters and selecting the final model. For inspections, the machine learning approach is highly successful and outperforms the previously existing defect estimation methods in software engineering by a factor of 4 in accuracy on the standard benchmark. The author's approach is well applicable in other contexts that are subject to small training data sets.

INTRODUCTION

This chapter describes a novel application of machine learning to an important estimation problem in software engineering – estimating the number of hidden defects in software artifacts. The number of defects is a software metric that is indispensable for guiding decisions about the software quality assurance during development. In engineering processes, management usually demands that a certain quality level be met for the products at each production step, for instance, that each product be 98 percent defect-free. Software can never be assumed defect-free. In order to assess whether additional quality assurance is required before a prescribed quality level is met, the software engineers must *reliably estimate* the

DOI: 10.4018/978-1-60566-766-9.ch025

defect content of their software document, be it code or some other intermediate software product, such as a requirements specification or a design artifact. This is a hard problem with substantial economic significance in software practice.

Software companies use defect classification schemes and possess broad practical experience about the kind of errors typically committed by their developers, but software engineering does not have a general theory available that would explain how, when, and where defects get inserted into software artifacts: Currently, no model explains the generating process of the software defects. As a consequence, any estimates must be based on (secondary) defect data that is observed during development and deployment of the software.

During development, defect data emerges mainly during software testing and software inspections. Software testing requires that executable code is available. A typical defect metric collected during testing is the number of defects detected in each test. Software reliability engineering uses this kind of data to predict the total number of defects (including the hidden ones) by extrapolating the cumulative number of defects found during testing. The rationale is that the quality of the code increases as testing progresses and defects get fixed, hence the growth of the defect curve should eventually flatten. Reliability models do not explain the defect generating process, nonetheless they can yield good estimates for the defect content of code.

For software documents other than code, *inspections* are the main quality assurance technique (Gilb & Graham, 1993). During an inspection, reviewers independently find defects in individual detection phases and group meetings in a structured way, using special reading techniques. Inspections are highly effective where testing is not possible, including textual specifications and design documents. Defect data that emerge during an inspection of a document include the number of defects found by each individual reviewer, or the number of different defects detected by the inspection team as a whole.

In principle, software reliability models can also be applied to inspections by replacing the code tests with the detection phases of the individual reviewers. Empirical studies show, though, that reliability models do not work well when transferred to inspections (see the next section). The models generally exhibit a high variation in their estimation error. In particular, the estimates can be extreme outliers without warning during the estimation procedure. As a result, there is a compelling need in software engineering for a reliable defect content estimation technique for inspections.

In this chapter, we view the defect content estimation problem for inspected documents as a *machine learning* problem. The goal is to learn the relationship between certain observable features of an inspection and the true number of defects in the inspected document, including the hidden ones. A typical feature that can be observed during an inspection is the total number of different defects detected by the inspection team; this feature provides a lower bound for the defect content of the document. Some inspection features may carry valuable *nonlinear* information about the defect content of the document; an example is the variation of the reviewers' performance during the inspection. We identify such features using the information-theoretic concept of mutual information. In order to be able to capture any nonlinear relationships between the features and the target, we use *neural networks* for learning from the data.

The use of machine learning in our context is motivated by the key observation that all existing defect estimation techniques for inspections restrict themselves solely to data coming from *just the current* inspection, that is, the inspection effected on the document whose defect content is to be estimated. Hence, the existing techniques miss the opportunity to exploit the knowledge gained in past inspections of similar documents: There is no learning involved in these techniques. However, the way in which the defect content problem presents itself – an estimation problem where a quantitative model must be

extracted from empirical data, without having an explanatory model available that generates the data – already suggests the application of machine learning techniques.

From a software engineering perspective, our machine learning approach is *novel and highly successful* for the defect content estimation problem (Padberg & Ragg, 2004). On the standard benchmark consisting of 16 inspections carried out at NASA, the defect estimates previously obtained were much too inaccurate to be useable in practice (Section 7.2). In contrast, neural networks outperform the best previous technique by a factor of 4 in accuracy on the benchmark (Section 7.1). Exploiting the nonlinear information carried by certain features proves to be essential for the quality of the estimates and justifies using nonlinear models (Section 7.3). The high quality of the machine learning estimates on the inspection benchmark could well be considered a breakthrough for the defect content estimation problem, made possible by machine learning.

From a machine learning perspective, the main focus of this chapter is on *learning from small data sets*. There seems to be a widespread belief among users that neural network techniques are applicable only when plenty of data is available for training – yet, this opinion is not quite correct. The ultimate goal of training is to achieve a good generalization ability of the resulting model, and this is possible for small data sets, too. The key constraint is to avoid overfitting the sparse data. To this end, a bundle of established machine learning techniques are available. To support transferring our approach to other settings, we present a detailed description of each technique in this chapter: We use automatic feature selection (Section 5.3) to balance the dimension of the input space against the number of patterns available for training. During training, we use an error function with a regularization term (Section 6.2) to avoid large weights or a large number of hidden units. A consistent application of Bayesian techniques permits us to use the whole data for training, instead of reserving patterns for cross-validation of hyperparameters (Section 6.3) or selecting models (Section 6.4). In addition, we present a detailed analysis of the impact that the dimension of the feature vector and size of the hidden layer have on the performance of the neural network models (Section 7.4).

We would like to point out that small data sets can easily occur in other machine learning applications as well. For example, it may be necessary to subdivide an inhomogeneous data set into different classes and learn a separate model for each class; in this case, each class might contain only a small number of patterns. A similar argument applies when launching a machine learning approach in a domain where only few data are available at first. Therefore, our approach is highly relevant for other machine learning applications.

STATE OF DEFECT CONTENT ESTIMATION FOR INSPECTIONS

Reliability Models

Software reliability engineering offers several models for estimating the true number of defects contained in *code*. The underlying theme is that the quality of the code increases over time as software testing progresses and defects get found and fixed. Hence, it should be possible to predict the true number of defects (including the hidden ones) by extrapolating the growth of this curve. Technically, reliability modeling formalizes the software tests performed during development as a stochastic process that is parameterized by the number of defects initially contained in the code (Xie, 1991; Kan, 1995; Fenton & Pfleeger, 1997). The results observed in a concrete test series then are used to estimate the true number

of defects by applying suitable statistical techniques. The reliability models differ from each other with respect to their assumptions about the distribution of defects over the code, the particular shape of the growth curve, and the probabilities that some test will find a defect. For code, reliability models often yield good estimates for the defect content (Padberg, 2003).

Software researchers have applied the most common class of reliability models, capture-recapture models, to software inspections, see (Eick & Loader, 1992; Vander Wiel & Votta, 1993; Briand & El-Emam, 2000). Capture-recapture models were developed in biology to estimate the size of an animal population by repeated captures of samples. The captures correspond to the stages in the formal stochastic process, and the overlap in repeated captures allows to estimate the unknown size of the animal population. In an inspection setting, the reviewers replace the individual caputures in the series, and a defect detected by more than one reviewer corresponds to a repeated “capture.” The input for the capture-recapture models, such as the number of defects detected by each reviewer, can be easily computed from the raw inspection data. Note that only data about the inspection of the document to be estimated enters the capture-recapture model and, hence, the defect estimate; no historical data from past inspections is used for the estimation model.

Comprehensive empirical studies analyze the performance of capture-recapture models in an inspection context (Briand & El-Emam, 2000; Biffl & Grossmann, 2001). For inspections, capture-recapture models exhibit both large estimation errors (including extreme outliers) and a high variation in the error; in addition, they show a tendency to underestimate the target value. Therefore, conventional reliability models are useless in inspection estimation practice.

Curve-Fitting Methods

Some researchers have experimented with curve-fitting methods to estimate the defect content for inspections (Runeson & Wohlin, 1998). For example, the detection profile method computes for each defect the number of reviewers who have detected that particular defect from the raw inspection data. These numbers are sorted in decreasing order; then, an exponential curve is fitted through the sorted data points using linear regression on the log values. The defect content estimate is defined as the x-value of the intersection point of the exponential curve with the horizontal line $y = 0.5$. The rationale is that some reviewers perform better than others, and some defects are more easily found than others; hence, if a significant number of defects was detected by just one reviewer, this is taken as a sign that most defects have actually been found during the inspection. Similar to the capture-recapture methods, no historical data from previous inspections enters the defect estimate.

On the original data sets, the performance of the detection profile method was acceptable (Runeson & Wohlin, 1998). On independent validation data sets, though, this method performs worse than the capture-recapture methods, both with respect to its mean and max absolute error (Briand & El-Emam, 1998; Biffl & Grossmann, 2001). Even when accepting the method’s underlying rationale, the correspondence between singly-detected defects and the total number of defects is not reflected well by the curve-fitting process. In addition, the choice of the threshold value is arbitrary and not deducible within the method. As a result, the curve-fitting methods are much too unreliable to be used in inspection estimation practice.

Reviewer Profile-Based Methods

For small inspection teams with only two or three reviewers, capture-recapture and curve-fitting methods raise additional problems. With a small team, the result of the inspection is especially sensitive to the detection results of the individual reviewers (Briand & El-Emam, 1998; El-Emam & Laitenberger, 2001). In addition, chances are that there is no detection overlap among the reviewers, that is, only singly-detected defects emerge during the inspection. Due to their particular mathematical structure, capture-recapture models then yield no estimate at all (Briand & El-Emam, 2000; Padberg, 2003).

As an alternative approach, software researchers have used the past performance of the reviewers for estimating the defect content of an inspection (Wohlin & Petersson, 2001). The performance of the reviewers is measured by their average historical detection effectiveness (number of defects detected, divided by the total number of defects contained in the document); the average is taken on an individual or inspection team basis. The rationale is simple: It is assumed that if a reviewer detected an average of, say, 20 percent of the defects contained in the documents during past inspections, he will detect a similar percentage of all defects in the next inspection.

The reviewer profile-based approach does not take into account that the detection results of reviewers typically vary largely from inspection to inspection, as they depend on various characteristics of the document and human factors. Thus, it seems unlikely that a reviewer's past performance is a good indicator for his future performance. Detailed data about the characteristics of the inspection itself, such as the total effectiveness of the inspection or the detection overlap among the reviewers, do not enter the estimates in this approach. The reviewer profile-based method is also problematic from an ethical point of view, because the inspection results cannot be anonymized; such kind of individual performance control is known to have a negative impact on the developer motivation and performance. As a result, the estimation accuracy of the reviewer profile-based approach proves to be weak, showing a high error and a large error variation, similar to the capture-recapture methods (Wohlin & Petersson, 2001).

INPUT DATA

As the empirical basis for learning the relationship between an inspection's features and the target value, we must collect data from as many past inspections as possible. For each inspection in the empirical data base, we must know the proper raw data to compute all interesting candidate features; in addition, we must know the target value for each inspection in the data base, that is, the total number of defects that were contained in the inspected document.

The candidate features of an inspection can be derived from the *zero-one-matrix* of the inspection. This matrix shows which reviewer detected which defect in the inspection. Assuming proper bookkeeping in the company's inspection process, the zero-one-matrix gets compiled by the inspection leader at the end of an inspection to facilitate computing certain inspection efficiency metrics, such as the reviewer detection overlap. Hence, collecting this raw data for our estimation purposes does not impose an additional cost on the inspection process.

Determining the total number of defects in an inspected document requires more effort. For one thing, the *true* number of all defects in a software document is never known for sure. For example, even after comprehensive testing there still might be defects in a piece of code that went undetected; the same applies to other software documents. Secondly, defects in a software document usually show up at

Table 1.

defect / reviewer	#1	#2	#3	#4	#5	#6	#7	#8	Σ
A	0	1	1	1	1	0	0	1	5
B	0	0	0	1	0	0	0	1	2
C	1	1	0	1	0	0	1	0	4
D	0	0	0	1	0	0	1	0	2

several different times in the development process. In order to count the number of defects detected in a document during the whole development and maintenance cycle, the company must employ a *defect tracking system* that allows the software managers to trace each defect back to the documents where the defect was introduced. Assuming that such a tracking system is in place, the true number of defects in a document can be *approximated* by adding up all defects that were detected in this document after its inspection, including the maintenance phase. This approximate number gives a lower bound for the true number of defects and is sufficient in practice: The *relevant* defects that escaped development usually are quickly uncovered by the customers during the software's use.

Prior to learning an estimation model, the inspections in the empirical data base should be *sorted* according to suitably chosen inspection meta-data. In particular, the data set should be sorted according to the document type. In general, documents of different type exhibit different defect patterns and densities. Thus, it would make little sense to use, say, source code inspection data to estimate the defects in a requirements document. Additionally, the sorting may consider the document complexity, or, whether an explicit reading technique was used, such as checklist-based reading. Note that the plain size of a document (measured in lines of code or lines of text) often is an inappropriate metric for assessing a document's complexity. In particular, the difficulty of understanding a piece of code depends more on how involved the code is and how familiar the reviewer is with the domain than on the code's mere size. Similarly, different reading techniques may exhibit a varying defect detection effectiveness. The estimation model should best be learned from inspections having a type similar to the inspection that is to be estimated.

BENCHMARK DATA SET

For illustration and validation purposes, we use a well-known inspection benchmark as our running example in this chapter. The benchmark consists of 16 inspections that were conducted during controlled experiments in 1994 and 1995 at the NASA Software Engineering Laboratory SEL.

The documents were mildly structured, textual requirements specifications. Two documents came from NASA, two documents were generic (the specification of a parking garage control system, and the specification of an automated teller machine). To account for the varying size of the documents (between 16 and 27 pages), the reviewers were given enough time to finish each inspection. The size of the inspection team varied between six and eight reviewers; in most cases, there were six reviewers. Half of the inspections were conducted using ad-hoc reading, the other half applied an explicit reading technique (perspective-based reading). The original goal of the experiments was to assess the effectiveness of the reading techniques. The reviewers were software professionals from the NASA SEL.

For each inspection in the benchmark, the zero-one matrix is on record. Table 1 shows a fraction of the zero-one matrix for one of the inspections from the data base.

The requirements documents that served as a basis for the experiments originally were all considered defect-free: The NASA documents had been checked several times before at NASA, and the generic documents had been checked thoroughly in earlier experiments. For the purpose of the 1994/95 experiments, a number of typical defects were *seeded* into the documents to have full control over the target value. This way, the true number of defects is known exactly for each inspection in the benchmark.

PREPROCESSING THE TRAINING DATA

Sorting the Inspections

As the first step in preprocessing the input data, the inspections in the data base should be sorted according to suitable inspection meta-data, as described in the section on Input Data. With respect to the document type, our benchmark data set already is sufficiently homogeneous. Given the small size of the data set, we do not sort and split the data set any further, f.e., according to the particular reading technique. Instead, we use the full set of 16 inspections.

Finding Candidate Features

The zero-one matrix itself is not suitable as input for learning. The matrix is too large (see the next subsection) and its dimensions vary from inspection to inspection. Therefore, we must compute *features* from the matrix, f.e., the total number of different defects detected by the inspection team. To make different inspections comparable, the candidates should be independent of parameters that tend to vary between inspections, such as the inspection team size, the number of detected defects, or the document size. Apart from that, the features should carry maximal information about the characteristics of the inspection.

One can think of a number of possible candidate features of an inspection. How to choose “good” candidate features that carry relevant information about the target is *not* a mathematical problem, though. Whether a particular choice is good or not can only be judged by assessing the predictive performance of the resulting model.

For our learning approach, we generate a set of five candidate features from the zero-one matrix of an inspection:

- the total number tdd of different defects detected by the inspection team;
- the average ave, maximum max, and minimum min number of defects detected by the individual reviewers;
- the standard deviation std of the number of defects detected by the individual reviewers.

The candidate features are straightforward and measure the overall effectiveness of an inspection, the performance of the individual reviewers in the inspection, and the variation in the reviewers’ performance.

For example, the features derived from the zero-one matrix in the table above are as follows: tdd = 7; ave = 3.25; max = 5; min = 2; std = 1.5.

Table 2.

dimension / features	samples / patterns
1	4
2	19
3	67
4	223
5	768
6	2790

Feature Selection

Prior to entering any training process, a suitable subset of the candidate features must be selected as input for the models. In general, the dimension of the feature vector must be *reduced* to fit the size of the data set available for training. If the dimension of the input space is chosen too high, the training patterns will be sparse in input space (“empty space phenomenon”) and the resulting estimation models won’t be robust; small changes in the data could produce large variations in the models and estimates (Bishop, 1995). The required number of training patterns grows quickly with the dimension of the input space. Feature selection is especially important for small data sets. The larger the training data set, the more features can be used as input and, hence, the more information about the defect content may enter the estimate.

To determine how many features should be used for a given data set size, we use Silverman’s table (Silverman, 1986). Assuming that the relative mean squared error has to be less than 10 percent, Table 2 shows how many samples are required to estimate the value at the origin of the standard multivariate normal density of a particular dimension.

We use Silverman’s table the other way around as a practical rule of thumb to determine how many features should be used when the number of training patterns is given. For our benchmark data set containing 16 inspections, at most two features should be used as input for learning.

The selected features should carry as much information about the target as possible and should complement one another. To select the most promising features from our set of five candidates (tdd, min, max, ave, std), it would not make much sense to rank the features by a standard correlation analysis: Correlation analysis is a linear technique, and we aim at learning a nonlinear model. Instead, we use a ranking procedure that is based on the concept of *mutual information* (Cover & Thomas, 1991). The mutual information $MI(X;T)$ measures the degree of stochastic (thus, including nonlinear) dependence between two random vectors X and T . The mutual information is defined using the entropy H as

$$MI(X;T) = H(T) - H(T | X) = \iint p(x,t) \cdot \log \frac{p(x,t)}{p(x)p(t)}$$

If the mutual information value is high, X carries much information about T . To compute densities such as $p(x,t)$ from the raw inspection data, we use Epanechnikov kernel estimators (Silverman, 1986). Recall that at this stage, we merely aim at ranking the candidate features, not at computing the model.

As the first step in the feature selection procedure, we select that feature F among the candidates which carries the most information about the target, the defect content num:

$$\max_F \text{MI}(F; \text{num})$$

For the benchmark data set, the feature tdd (the total number of different defects detected by the inspection team) maximizes the mutual information with the target num and gets selected as the first feature in the input vector.

In the second step, we select that feature F among the remaining candidates which adds the most information about the target when used *in combination* with the previously selected feature:

$$\max_F \text{MI}((\text{tdd}, F); \text{num})$$

For the benchmark, the feature std (the standard deviation of the reviewers performance) – when taken together with the already selected feature tdd – maximizes the mutual information with the target num and gets selected as the second feature in the input vector.

The procedure continues this way, always selecting that feature next which adds the most information about the target when combined with all the previously selected features. In the third step, the procedure aims at maximizing $\text{MI}((\text{tdd}, \text{std}, F); \text{num})$ and so on. In the end, a ranking of the candidate features emerges. For the benchmark, the ranking is $\text{tdd} > \text{std} > \text{max} > \text{min} > \text{ave}$. As input, we need only the features ranking first and second, namely, tdd and std.

Note that the ranking of the candidate features depends on the data – other data sets will likely yield a different ranking. Similarly, it is instructive to see that a (linear) correlation analysis would yield a different choice of features. The linear ranking is $\text{tdd} > \text{ave} > \text{min} > \text{max} > \text{std}$, see (Padberg & Ragg, 2004) for details. In particular, the feature std ranks last in the linear setting but second in the mutual information-based ranking. This is a strong indication that – for this data set – the variation in the reviewers' performance carries important *nonlinear* information about the defect content of the documents. The linear correlation between std and num is only weak. Thus, using a nonlinear model with tdd and std as input might improve the predictive performance considerably as compared to a linear model, motivating our neural network approach for learning.

LEARNING THE MODEL

Neural Networks

We use neural networks for learning models (Bishop, 1995). As the activation function of the hidden neurons, we apply the logistic function

$$f(x) = \frac{1}{1 + e^{-x}}$$

The output neuron applies the identity function, that is, it sums up the weighted output of the hidden neurons.

Neural networks provide much flexibility to adapt the complexity of the function approximator (that is, the model) to the size of the training data set. For example, for the rather small inspection benchmark data set, we train neural networks with just one hidden layer and a few hidden neurons in that layer, restricting the models to a moderate degree of nonlinearity; see the subsection on feature selection. For larger data sets, increasing the number of layers and neurons allows for more complex approximators.

Regularization

The goal of the training procedure is to capture the functional relationship between the features of an inspection and the defect content of the inspected document in a neural network model, exploiting the information available in the empirical data. To this end, it is *not* sufficient to learn a model that minimizes the error on the training data; it is well-known that merely trying to minimize the training error often leads to overfitting the empirical data, including any noise in the data. It is more important that the model *generalizes*, that is, yields good estimates on previously unseen input. Therefore, it is necessary to *balance* the training error against the complexity of the model (Bishop, 1995, chapter 9). To achieve such a balance, we train the neural networks such that they minimize the *regularized* error

$$E = \beta \cdot E_D + \alpha \cdot E_R$$

The term E_D is the usual mean squared error on the training data. The regularization term

$$E_R = \frac{1}{2} \cdot \sum_k w_k^2$$

is known as the weight decay; the w_k denote the weights (and biases) in the neural network. The weight decay is a measure for the model complexity. This way, large weights or a large number of hidden units get penalized during training. Still, the networks can contain large weights after training, but only when the training data contains strong evidence for this.

The factors α and β are additional model parameters that are determined during training, see the next subsection.

Bayesian Training

During training, the weights of the neural network iteratively get adjusted in such a way that the error function attains a (local) minimum on the training data. This minimization usually is done by backpropagation. Our training procedure is more involved, though, because we also must determine the additional parameters α and β during training. We alternate between the following two steps:

- Fix α and β . Optimize the weights w_k using the gradient descent algorithm Rprop (Riedmiller, 1994). Compute all gradients of the weights w_k with respect to the regularized error function E by backpropagation (Rumelhart & Hinton, 1986).

- Update α and β according to the update rule: $\alpha^{new} = \frac{\gamma}{2 \cdot E_R}$ and $\beta^{new} = \frac{N - \gamma}{2 \cdot E_D}$ where $\gamma = \sum_{i=1}^W \frac{\lambda_i}{\lambda_i + \alpha^{old}}$

Here, N denotes the number of training patterns, the λ_i are the eigenvalues of the Hessian matrix of the error function E_D on the training data, and W is the number of weights (including biases) in the network. The update rule is based on Bayesian methods, see below.

We start with a random choice for α and β , and keep alternating between the two steps until a fixed of number of training cycles (typically, 1000 epochs) has been reached.

It is common practice to determine model parameters (hyperparameters) such as α and β by cross-validation on separate data. Since the benchmark data set is very small, we don't want to cut off part of it for cross-validation. Instead, we take a Bayesian approach (MacKay, 1992; Ragg & Menzel, 2002). The idea is to maximize during training the *posterior probability* $P(\alpha, \beta | D)$ of the parameters given the training data. The posterior probability can be computed from the *evidence* $P(D | \alpha, \beta)$ using Bayes rule:

$$P(\alpha, \beta | D) = \frac{P(D | \alpha, \beta) \cdot p(\alpha, \beta)}{p(D)}$$

Since nothing is known about suitable values for α and β at the beginning of training, we choose a non-informative prior $p(\alpha, \beta)$ for the joint parameter distribution, basically assigning equal weight to all possible values. Maximizing the posterior probability then amounts to maximizing the evidence. To compute the maximum of the evidence, one assumes a Gaussian form of the prior $p(\mathbf{w})$ for the network weights and a Gaussian noise for the target, then approximates the regularized error function by its Taylor expansion around its minimum, computes the derivative of the logarithm of the evidence with respect to α and β , and arrives at two specific equations that hold for the maximum; these equations turn into the update rule. For more details, see (Bishop, 1995, chapter 10.4).

For data sets where the number of training patterns is much larger than the size of the network (N much larger than W), the update rules can be simplified (Bishop, 1995, chapter 10.4):

$$\alpha^{new} = \frac{W}{2 \cdot E_R} \text{ and } \alpha^{new} = \frac{W}{2 \cdot E_R}, \quad \beta^{new} = \frac{N}{2 \cdot E_D} \text{ and } \beta^{new} = \frac{N}{2 \cdot E_D}.$$

This form does not require the Hessian of the error function and is much faster to compute.

The learning curve for the neural networks reflect the two-step training procedure. Whenever the parameters α and β are fixed, the training error goes down quickly with gradient descent. Each time when α and β are updated the training error increases again. Overall, the training error decreases.

In software engineering, empirical data sets typically are small, as is f.e. the inspection benchmark. Therefore, it is crucial to exploit as much information as possible when learning the models. Bayesian methods allow us to avoid consuming part of the data for cross-validation of hyperparameters and leave the full data set for training.

Model Selection

Training a neural network is a nonlinear, iterative optimization process that bears the risk of getting caught in a *local* minimum of the error function instead of achieving the global minimum. To avoid obtaining a suboptimal model, many different networks with randomly chosen initial weights are trained on the input. This leaves the task of selecting the “best” from all trained networks at the end.

In addition to training networks with different initial weights, we also train networks with different topologies to allow for varying degrees of nonlinearity in the model. In general, this involves varying the number of hidden layers and neurons. In case of the small benchmark data set, we restrict our networks to one hidden layer in order to avoid overfitting the data; then, we vary the number of neurons in the hidden layer between 1 and 10, trying models with a moderate degree of nonlinearity.

The standard approach for selecting the best model from a variety of neural networks is to evaluate the predictive performance of each network on an independent data set (cross-validation). If no additional data set is available, one must split the given data into two parts, one used for training, the other used exclusively for cross-validation. This situation is similar to the determination of the model hyperparameters α and β as described in the previous subsection. To save more data for training, we again apply a *Bayesian approach* for selecting the final model instead of cross-validation.

The idea of the Bayesian model selection approach is to maximize the posterior probability $P(\mathbf{w}, \alpha, \beta, h | D)$ where \mathbf{w} is the weight vector of the network and h stands for its topology (layers, neurons, and their connections). By assigning the same prior to all networks h and integrating out the parameters α , β , and \mathbf{w} , the posterior probability can be approximated by the *model evidence* $P(D | h)$ of the trained networks. For the logarithm of the evidence, a closed formula can be given (Bishop, 1995, chapter 10.6):

$$\begin{aligned} \ln(P(D | h)) = & -\alpha \cdot E_R - \beta \cdot E_D - \frac{1}{2} \cdot \ln |\mathbf{A}| + \\ & + \frac{W}{2} \cdot \ln \alpha + \frac{N}{2} \cdot \ln \beta + \ln \tilde{h} + \frac{1}{2} \ln \frac{2}{\gamma} + \frac{1}{2} \ln \frac{2}{N - \gamma} \end{aligned}$$

Here, α , β , E_R , E_D , W , N , and γ denote the same quantities as before. In addition, the formula involves the determinant $|\mathbf{A}|$ of the Hessian matrix of the regularized error function. The quantity \tilde{h} represents the topology of the network and is defined as $\tilde{h} = k^m m!$ where k is the number of layers and m is the number of hidden units in the network.

The selection of the final model now is made according to the model evidence of the networks:

- For each of the trained networks, we compute the (logarithm of) its model evidence.
- Then, we fix the network topology, compute the *average* evidence for all networks with that particular topology, and select the topology with the best (that is, largest) average evidence.
- Finally, from all networks with the best topology we select the network with the best (that is, largest) evidence as the final model.

This selection procedure is justified by practical experience which shows that the model evidence correlates well with the generalization ability of the model, as long as the neural network is reasonably small relative to the number of training patterns (no overfitting). In addition, practical experience shows

that first choosing the topology with the best evidence and then the final model yields more robust models with a better generalization ability, as compared to immediately selecting the model with the best evidence from all trained networks.

As an example, suppose we leave out the second inspection from the benchmark as a test data point and keep the remaining 15 inspections for training. We vary the number of hidden units (topology) from 1 to 10. For each topology, we train 50 networks using the alternating training procedure from the previous subsection. The topology with two hidden units exhibits the best average log evidence (-22.5); closest to this are the topology with one resp. three hidden units (-22.6 and -23.6). The best among all networks with two hidden units has a log evidence of -20.0 and is selected as the final model. The corresponding test error on the left-out second data point is about 3 percent.

We note that the best models with one, resp., three hidden units show a similar test performance. Given such a small data set, one hidden unit provides enough flexibility to capture the data, whereas three units do not improve the result but increase the risk of overfitting the data. This picture in general may change for increasingly larger data sets, though.

The Bayesian approach gives us a method for selecting the final model that is based on the results of the training procedure alone, and that does not require any cross-validation data. As a result, all data remains available for training. This is a great advantage when the data sets tend to be small.

VALIDATION

Jackknife Results

We perform a *jackknife validation* (“leave one out”) of our machine learning approach on the benchmark data set. First, for each inspection in the benchmark we compute the features tdd and std as input for the networks, see Table 3. The target is the number of defects num in a particular document (including all defects detected in the inspection and all still hidden defects). Recall that the target is known exactly in each case because the defects had been seeded into the (otherwise defect-free) documents.

Next, we leave out an inspection from the data set that is to be used as the test pattern. We take the remaining 15 inspections as the training patterns and compute a neural network model using the Bayesian training and model selection procedure described in the preceding section. Finally, with the resulting model we estimate the test pattern and get a certain estimation error (test error).

For example, suppose we leave out inspection no. 5 from the data set and learn a model using the remaining inspections no. 1 – 4 and 6 – 16 as the training data. We get a model that yields an estimate of 29.37 (rounded to 29) for the defect content of document no. 5 when prompted with the test pattern (20, 1.7). Since the target value is 28, the model overestimates the defect content of document no. 5 by only 4 percent.

Table 3.

inspection	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
feature tdd	23	20	24	22	20	17	24	21	10	6	15	15	6	9	14	13
feature std	2.4	3.3	2.9	2.7	1.7	2.3	2.5	4.0	1.5	0.9	3.0	3.4	1.4	1.9	4.5	3.8
target num	30	30	30	30	28	28	28	28	18	18	15	15	15	15	15	15

Systematically varying the left-out inspection over the whole benchmark, we end up with 16 different models each of which is employed for estimating the defect content of the corresponding, left-out test pattern. Table 4 shows these estimates and the test errors:

On the benchmark, the mean of the absolute relative errors (“jackknife error”) is only 5 percent. The maximum absolute error is about 17 percent; no outliers occur. As a result, on this data set machine learning yields *highly reliable estimates* for the defect content. Machine learning outperforms the best previous technique by a factor of 4 in accuracy on the benchmark, see below.

Comparison with Capture-Recapture Models

On the inspection benchmark, capture-recapture estimates exhibit a strong tendency to underestimate the defect content of the documents. For example, the best capture-recapture method Mt(MLE) yields the estimates 24—22—25—23—22—21—24—21—11—6—15—15—8—10—14—13 for the 16 inspections in the benchmark (Briand & El-Emam, 2000). The mean absolute error is 24 percent, with a max absolute error of 67 percent. The results for other capture-recapture models are very similar (Briand & El-Emam, 2000); the detection profile method (Runeson & Wohlin, 1998) performs even worse with a mean absolute error of 36 percent and a max absolute error of 113 percent (Padberg, 2002). Clearly, both the error and its variation are too high for these methods to be of any practical value when estimating inspected documents.

To understand why the capture-recapture estimates are so far off for inspections, it is instructive to study how the capture-recapture estimates for code typically depend on the *length* of a test series. Early during the test series, the estimates fluctuate significantly, exhibiting a “zig-zag” pattern. Each new test brings additional information about the defect content of the code and has an according impact on the next estimate. The first few estimates also are gross underestimates. After about ten to fifteen tests in the series, the estimates stabilize. This behavior is not really a problem for code, where a test series typically comprises several dozen tests. When applying the capture-recapture methods to inspections, though, it becomes obvious why the estimates fail: Typically, only three or four reviewers participate in an inspection, and this number of is too small for a capture-recapture estimate to stabilize.

Table 4.

model	1	2	3	4	5	6	7	8
target	30	30	30	30	28	28	28	28
estimate	29.04	28.94	29.05	29.03	29.37	29.35	29.40	23.90
rounded	29	29	29	29	29	29	29	24
test error %	-3.3	-3.3	-3.3	-3.3	3.6	3.6	3.6	-14.3
model	9	10	11	12	13	14	15	16
target	18	18	15	15	15	15	15	15
estimate	15.41	14.91	17.38	15.49	15.34	15.35	15.35	15.34
rounded	15	15	17	15	15	15	15	15
test error %	-16.7	-16.7	13.3	0	0	0	0	0

Comparison with Linear Models

Applying a nonlinear, highly iterative regression technique such as neural networks causes a considerable computational effort. Therefore, it is natural to check whether a standard *linear* regression would yield acceptable results at much lower cost.

In the subsection on Feature Selection, we noted that a linear correlation analysis gives the feature ranking $tdd > ave > min > max > std$, see (Padberg & Ragg, 2004). For the same arguments as in the neural network case, it doesn't make much sense to use more than two features as input for the linear regression. Using tdd and ave as the features, the jackknife error for the linear models is 11 percent, about twice as large than for the neural networks (Padberg & Ragg, 2004). Even worse, the absolute error for the linear models ranges as high as 40 percent, compared to 17 percent for the neural networks. Although the linear models perform much better than the capture-recapture models, this high and unforeseeable variation in the error makes the linear models useless from a practitioner's viewpoint and provides a clear motivation for using some nonlinear technique such as neural networks.

Even when restricting the input vector to only one feature (tdd), the neural networks outperform the linear models. The jackknife error for linear regression in this case is 19 percent, for the nonlinear models 9 percent (Padberg & Ragg, 2004). Obviously, the training data contains some nonlinear component other than std that cannot be exploited by linear regression.

Analysis of the Results

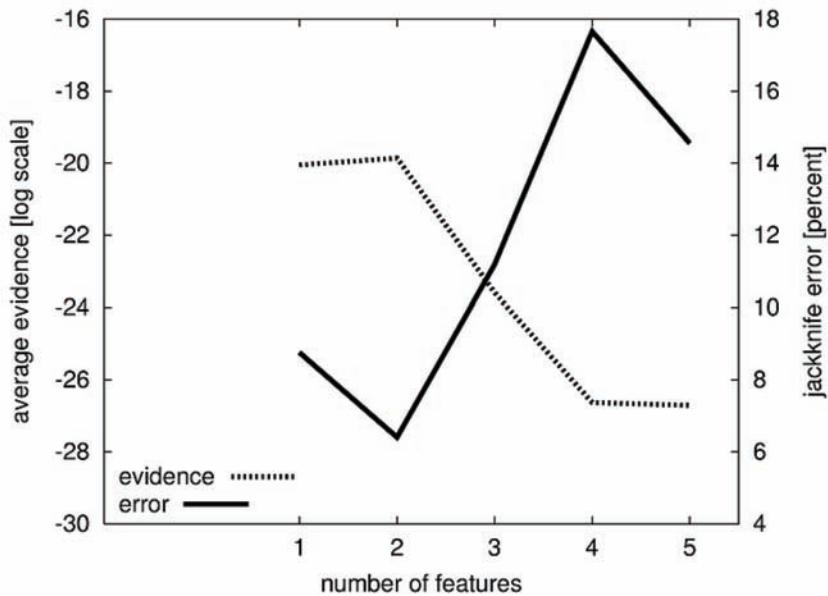
Looking back at the good results for the benchmark data, it is well worth analyzing why the machine learning approach works for this particular estimation problem.

For the technical side, it was crucial to train only small networks on just a few features in order to achieve a good generalization ability of the resulting models. There is only a limited amount of information in any given data; trying to squeeze the data too much inevitably leads to overfitting the data and, hence, to unfeasible or even bad estimates. The smaller the data set, the more relevant this fact is. In software engineering there are a number of applications of neural networks where such an overfitting occurred (Karunanithi & Whitley, 1992; Khoshgoftaar & Pandya, 1992; Khoshgoftaar & Szabo, 1996). The risk of overfitting can be reduced by *downsizing* the number of features relative to the number of training patterns (f.e., based on Silverman's rule), and by using a regularized error function that penalizes large weights in the networks. Splitting an already small data set for the purpose of cross-validation or model selection can be avoided by applying Bayesian techniques.

Figure 1 shows on the benchmark that the performance of the models *decreases* when features are added as input, see the solid line for the jackknife error. The features were added according to the ranking $tdd > std > max > min > ave$ that was computed with mutual information. For each input vector, the whole training and model selection procedure was performed, then the jackknife validation. The dotted line shows the average evidence of the 16 final models for each input vector. There is an inverse correspondence between the jackknife error and the average evidence; the jackknife error is minimal and the average evidence maximal when using two features, tdd and std . After the event, figure 1 justifies experimentally that we used no more than two features as input.

Similarly, figure 2 shows (for two input features) that adding units to the hidden layer tends to *decrease* the performance of the models, see the solid lines for the jackknife errors. Instead of the full model selection procedure, for each of the 16 validation cases we selected the model with the best evidence among

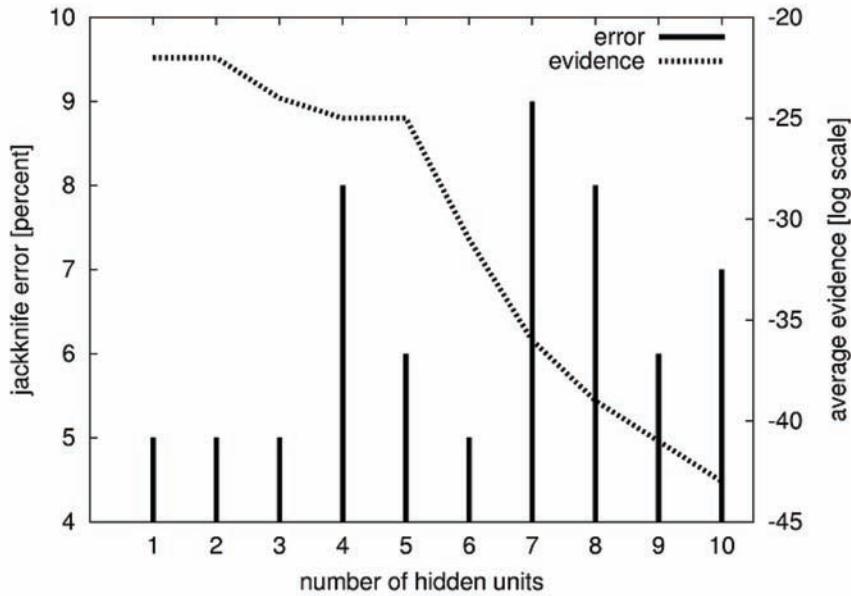
Figure 1. Impact of the number of features on the jackknife error and model evidence



the networks with the given number of hidden units as the final model. Accordingly, the dotted line shows the average evidence of the best 16 models with the given number of hidden units. The jackknife error is smallest with one to three hidden units; at the same time, the average model evidence is highest with one or two hidden units. The results indicate that networks with too many units might try to learn a fake relationship from the training patterns that is not present in the generating process.

For the software engineering side, it is comprehensible why the feature selection process yields the two particular features tdd and std as input for the networks. Clearly, the number tdd of different defects detected by the inspection team is a lower bound for the true number of defects in the document and is important input to back up the estimates. The feature std is important supplementary input that allows the models to distinguish between different kinds of inspections that otherwise might exhibit the same tdd value. For example, it is not unusual that in some inspection a few reviewers detect a large number of all defects while the other reviewers detect only a few defects; this often happens when some reviewers are experts in the application domain. Yet, there are other inspections where each reviewer detects about the same number of defects. Experience shows that these two cases of software inspection can point to significantly different counts of hidden defects. In the case of the benchmark, the feature std seems to primarily support a decision for low tdd values. If few defects have been detected by the team (no more than, say, the median of 16), a small std value indicates that tdd is significantly off the true defect count, whereas a large std value indicates that tdd is already close to the true value; see the data for the inspections numbered 9 – 16, and 6.

Figure 2. Impact of the number of hidden units on the jackknife error and model evidence



CONCLUSION AND FUTURE TRENDS

In this chapter, we adopted a machine learning approach to estimate the number of hidden defects in software requirements specifications. We used neural networks for learning. The input features were computed from the results of an inspection of the document. Besides testing, inspections are the main quality assurance technique in software practice today.

For software inspections, the empirical data sets typically are small. To achieve a good generalization ability of the models it was crucial to avoid overfitting the sparse data and to make optimal use of the available data during training. To avoid overfitting, we adapted the complexity of the models to the amount of training data. By applying Bayesian techniques during training and for model selection, we eliminated the need for putting aside part of the data as cross-validation data; this way, all empirical inspection data remains available for training.

The estimation results on the standard inspection benchmark are encouraging: Machine learning outperforms the existing techniques for estimating the defect content by a factor of 4 to 6 in accuracy. Using the proper techniques, machine learning can be highly successful even for small data sets. We expect that this observation holds in other problem contexts as well.

Defect estimation for software documents is an active area in software engineering research. Open questions addressed in current research include defect content models for novel development processes such as Extreme Programming (Sherriff & Nagappan, 2005) and techniques for automatic defect localization in less-structured software documents, such as informal requirements (Song & Shepperd, 2006). Given the kind of data emerging in such problems, software researchers sooner or later turn to the field of machine learning for the necessary methods to automatically detect patterns in their data.

In the software industry, there is an ongoing strong trend towards *certified* engineering and develop-

ment processes, see the capability maturity models CMM and CMMI (Ahern & Clouse, 2006). Tracking and evaluating the effectiveness of the quality assurance measures taken during development, such as testing and inspections, is an indispensable element of the higher maturity levels. Hence, industry is setting up systems for backtracking defects over the whole development cycle, and is compiling inspection data sets for validation and experimentation purposes. Clearly, *automatic and reliable* defect content estimation techniques – such as the machine learning approach presented in this chapter – play a key role in achieving higher maturity levels and, hence, will gain added importance in the future.

REFERENCES

- Ahern, D. M., Clouse, A., & Turner, R. (2006). *CMMI distilled*. Reading, MA: Addison-Wesley.
- Basili, V. R., Green, S., Laitenberger, O., Lanubile, F., Shull, F., Sørumgård, S., & Zelkowitz, M. V. (1996). The empirical investigation of perspective-based reading. *Empirical Software Engineering*, 1(2), 133–164. doi:10.1007/BF00368702
- Biffl, S., & Grossmann, W. (2001). Evaluating the accuracy of defect estimation models based on inspection data from two inspection cycles. In *Proceedings of the 23rd International Conference on Software Engineering* (pp. 145-154).
- Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford, UK: Oxford Press.
- Briand, L. C., El-Emam, K., & Freimut, B. G. (1998). A comparison and integration of capture-recapture models and the detection profile method. In *Proceedings of the 9th International Symposium on Software Reliability Engineering* (pp. 32-41).
- Briand, L. C., El-Emam, K., Freimut, B. G., & Laitenberger, O. (2000). A comprehensive evaluation of capture-recapture models for estimating software defect content. *IEEE Transactions on Software Engineering*, 26(6), 518–540. doi:10.1109/32.852741
- Cover, T. M., & Thomas, J. A. (1991). *Elements of information theory*. New York: Wiley.
- Eick, S. G., Loader, C. R., Long, M. D., Votta, L. G., & Vander Wiel, S. A. (1992). Estimating software fault content before coding. In *Proceedings of the 14th International Conference on Software Engineering* (pp. 59-65).
- El-Emam, K., & Laitenberger, O. (2001). Evaluating capture-recapture models with two inspectors. *IEEE Transactions on Software Engineering*, 27(9), 851–864. doi:10.1109/32.950319
- Fenton, N. E., & Pfleeger, S. L. (1997). *Software metrics*. Boston, MA: PWS Publishing.
- Gilb, T., & Graham, D. (1993). *Software inspection*. Reading MA: Addison-Wesley.
- Kan, S. (1995). *Metrics and models in software quality engineering*. Reading, MA: Addison-Wesley.
- Karunanithi, N., Whitley, D., & Malaiya, Y. K. (1992). Prediction of software reliability using connectionist models. *IEEE Transactions on Software Engineering*, 18(7), 563–574. doi:10.1109/32.148475

- Khoshgoftaar, T. M., Pandya, A. S., & More, H. B. (1992). A neural network approach for predicting software development faults. In *Proceeding of the 3rd International Symposium on Software Reliability Engineering* (pp. 83-89).
- Khoshgoftaar, T. M., & Szabo, R. M. (1996). Using neural networks to predict software faults during testing. *IEEE Transactions on Reliability*, 45(3), 456–462. doi:10.1109/24.537016
- MacKay, D. J. C. (1992). A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3), 448–472. doi:10.1162/neco.1992.4.3.448
- Padberg, F. (2002). Empirical interval estimates for the defect content after an inspection. In *Proceedings of the 22nd International Conference on Software Engineering* (pp. 58-68).
- Padberg, F. (2003). Maximum likelihood estimates for the hypergeometric software reliability model. *International Journal of Reliability Quality and Safety Engineering*, 10(1), 41–53. doi:10.1142/S0218539303000981
- Padberg, F., Ragg, T., & Schoknecht, R. (2004). Using machine learning for estimating the defect content after an inspection. *IEEE Transactions on Software Engineering*, 30(1), 17–28. doi:10.1109/TSE.2004.1265733
- Ragg, T., Menzel, W., Baum, W., & Wigbers, M. (2002). Bayesian learning for sales rate prediction for thousands of retailers. *Neurocomputing*, 43, 127–144. doi:10.1016/S0925-2312(01)00624-5
- Riedmiller, M. (1994). Supervised learning in multilayer perceptrons – from backpropagation to adaptive learning techniques. *International Journal on Computer Standards and Interfaces*, 16, 265–278. doi:10.1016/0920-5489(94)90017-5
- Rumelhart, D. E., Hinton, G. E., & Williams, R. S. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533–536. doi:10.1038/323533a0
- Runeson, P., & Wohlin, C. (1998). Defect content estimations from review data. In *Proceedings of the 20th International Conference on Software Engineering* (pp. 400-409).
- Sherriff, M., Nagappan, N., Williams, L., & Vouk, M. (2005). Early estimation of defect density using an in-process Haskell metrics model. *Software Engineering Notes*, 30(4), 1–6. doi:10.1145/1082983.1083285
- Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. Boca Raton, FL: Chapman and Hall.
- Song, Q., Shepperd, M. J., Cartwright, M., & Mair, C. (2006). Software defect association mining and defect correction effort prediction. *IEEE Transactions on Software Engineering*, 32(2), 69–82. doi:10.1109/TSE.2006.1599417
- Vander Wiel, S. A., & Votta, L. G. (1993). Assessing software designs using capture-recapture methods. *IEEE Transactions on Software Engineering*, 19(11), 1045–1054. doi:10.1109/32.256852
- Wohlin, C., Petersson, H., Höst, M., & Runeson, P. (2001). Defect content estimation for two reviewers. In *Proceedings of the 12th International Symposium on Software Reliability Engineering* (pp. 340-345).

Xie, M. (1991). *Software reliability modelling*. Singapore: World Scientific Publishing.

KEY TERMS AND DEFINITIONS

Bayesian Learning: A learning technique that determines model parameters (such as the network weights) by maximizing the posterior probability of the parameters given the training data. The idea is that some parameter values are more consistent with the observed data than others. By Bayes' rule, maximizing the posterior probability amounts to maximizing the so-called model evidence, defined as the conditional probability of the training data given the model parameters. The evidence often can be approximated by some closed formula or by an update rule. Bayesian techniques render it possible to use all data for training instead of reserving patterns for cross-validation of parameters.

Defect Content: The number of defects contained in a software artifact. This number includes the defects that have not yet been found by quality assurance, that is, the hidden defects.

Empty Space Phenomenon: A case where the training patterns sparsely populate the input space. This occurs when the training data set is too small relative to the dimension of the input space. As a result, the models won't be robust and small changes in the data can produce large variations in the estimates.

Feature Selection: A systematic procedure that selects the most relevant features from a set of candidates as input for the networks. The goal is to select those features that together carry the most information about the target and to avoid that the input space gets too high-dimensional.

Generalization: The ability of a model to yield good estimates on previously unseen input.

Model Selection: A systematic procedure that selects the best neural network from a set of trained networks as the final model. The best network should show a small training error and at the same time a high ability to generalize.

Mutual Information: An entropy-based measure for the degree of stochastic dependence between two random vectors. If the mutual information value is high, the vectors carry much information about each other.

Overfitting: Learning a complicated function that matches the training data closely but fails to recognize the underlying process that generates the data. As a result of overfitting, the model performs poor on new input. Overfitting occurs when the training patterns are sparse in input space and/or the trained networks are too complex.

Regularization: Including a term in the error function such that the training process favours networks of moderate size and complexity, that is, networks with small weights and few hidden units. The goal is to avoid overfitting and support generalization.

Software Inspection: A core technique in software quality assurance where a group of reviewers independently and systematically examine software artifacts to find defects. Inspections are highly effective where software testing is not possible, in particular, for textual specifications and design documents.

Chapter 26

Machine Learning for Biometrics

Albert Ali Salah

Centre for Mathematics and Computer Science (CWI), The Netherlands

ABSTRACT

Biometrics aims at reliable and robust identification of humans from their personal traits, mainly for security and authentication purposes, but also for identifying and tracking the users of smarter applications. Frequently considered modalities are fingerprint, face, iris, palmprint and voice, but there are many other possible biometrics, including gait, ear image, retina, DNA, and even behaviours. This chapter presents a survey of machine learning methods used for biometrics applications, and identifies relevant research issues. The author focuses on three areas of interest: offline methods for biometric template construction and recognition, information fusion methods for integrating multiple biometrics to obtain robust results, and methods for dealing with temporal information. By introducing exemplary and influential machine learning approaches in the context of specific biometrics applications, the author hopes to provide the reader with the means to create novel machine learning solutions to challenging biometrics problems.

INTRODUCTION

Biometrics serves the identification of humans from their personal traits. As a rapidly growing field, it is initially pushed forward by a need for robust security and surveillance applications, but its potential as a natural and effortless means of identification also paved the way for a host of smart applications that automatically identify the user and provide customized services. With increasing awareness of its psychological, privacy-related and ethical aspects, there is no doubt that biometrics will continue to contribute to many technological solutions of our daily lives.

DOI: 10.4018/978-1-60566-766-9.ch026

The technology of biometrics relies on the input from a number of fields, starting with various kinds of sensors that are used to sample the biometric. Signal processing and pattern recognition methods are obviously relevant, as the acquired data need to be prepared for accurate and robust decisions. At its final stage, the system outputs a decision, which links the acquired and processed biometric trait to an identity. Algorithms and mathematical models developed by the machine learning community are frequently used in biometric systems to implement the decision function itself, but this is surely not the only contribution worth mentioning. We will show in this chapter that machine learning methods are useful in selecting appropriate feature representations that will facilitate the job of the decision function, in dealing with temporal information, and in fusing multi-modal information.

The goal of this chapter is to familiarize the machine learning researcher with the problems of biometrics, to show which techniques are employed to solve them, and what challenges are open in the field that may benefit from future machine learning applications. It is also intended to familiarize the biometrics researcher to the methods and ways of machine learning and its correct research methodology, and to provide the rudiments of a toolbox of machine learning. In the next section, we provide a general look at biometric systems, define some relevant terminology and broadly identify the research issues. The third section deals with learning and matching biometric templates. Since this is a very broad topic, a small number of demonstrative application examples are selected. The fourth section is on the use of dynamic information for biometric purposes, and it is followed by a section on the fusion of multiple biometrics. Before concluding, we give a machine learning perspective on how to evaluate a biometrics system.

A GENERAL LOOK AT BIOMETRIC SYSTEMS

The application area of biometrics with the grandest scale is in *border control*, typically an airport scenario. Within the national identity context, it is possible to conceive the storing and managing of the biometric information for the entire population of a country. A smaller scale application is *access control*, for instance securing the entrance of a building (*physical access control*) or securing a digital system (*logical access control*). In both applications, we have a *verification* (or *authentication*) problem, where the user has an identity claim, and a sampled biometric is checked against a stored biometric for similarity. In a sense, this is a one-class pattern classification problem.

The second important problem involves *identification*, where there is no identity claim, and the sampled biometric is matched against many stored *templates*. Checking passengers against a list of criminals, forensic applications, identification of individuals at a distance, or providing access in consumer products (e.g. fingerprint scanning on a laptop) would be typical applications. Depending on the application requirements, the problem may be sufficiently constrained to apply a discriminative approach.

The most important biometric modalities are fingerprint, face, iris, signature, palm print and voice. The biometric traits differ in their usability, convenience, security, and complexity. For providing access to a high-security facility, security is of primary importance, whereas a household appliance that identifies users via biometrics would strive to have maximum user convenience. Similarly, privacy can be a major determinant in the deployment of a particular biometric application. For this reason, a host of possible biometrics are considered for different applications, including DNA, gait, ear images, and even behaviours.

Basic Biometrics Terminology

In this section, we define frequently used terminology for research in biometrics. We have already defined identification and verification. The biometric *template* t_i (also called a *target*) is a recorded instance of the biometric trait during *enrollment* for subject i , and it is stored in a *gallery* of templates, denoted here with $T=\{t_i / i=1 \dots N\}$, where N is the number of *subjects* in the gallery. A *query* (or a *probe*) is a biometric recorded during the operation of the system, denoted with B , and it is matched against the templates in the gallery.

The accuracy of a biometric authentication system is measured by its *false acceptance rate (FAR)* and its *false rejection rate (FRR)*, where the former is the probability of accepting an *impostor*, and the latter is the probability of rejecting a *genuine* claim. More formally, assume $s(B, t)$ denotes a similarity function for the recorded biometric, and τ is a threshold for accepting an identity claim. The identity claim for subject d with a recorded biometric B is accepted if and only if:

$$s(B, t_d) > \tau. \quad (1)$$

The threshold τ determines the convenience-security trade-off of the system, and a higher threshold means that more subjects will be rejected, thus a lower FAR will be obtained at the expense of a higher FRR. Low FAR is required for high-security systems, whereas low FRR means greater user convenience. Commonly the authentication performance of a system is reported with its FRR at 0.001 FAR.

As a side remark, we caution the reader to pay attention to the impostor model used for reporting the performance of a biometrics system. The impostor claims can be based on *zero-cost* attacks, where there is no particular effort on the part of the impostor, or on *informed* attacks. The former approach is exemplified by using all the test samples in a database for one genuine and $N-1$ impostor claims, where N is the number of samples in the gallery. The latter approach is exemplified by creating expertly forged signatures for attacks on a signature-verification system, which is expected to produce a much higher FAR than the zero-cost approach.

The similarity function can be substituted by a distance function, in which case the threshold serves as an upper-bound for authentication. Especially in non-parametric models, the distance function plays an important role, and needs to be chosen carefully to suit the distribution of the data and the outliers. We briefly list frequently used distance functions here.

The most common distance function is the Euclidean distance:

$$D_{Euclidean}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (2)$$

where n denotes the biometric vector dimensionality. For a known distribution of biometric samples with covariance denoted by S , the squared Mahalanobis distance gives less weight to deviations in directions along which the data are scattered:

$$D_{Mahalanobis}^2(\mathbf{x}, \mathbf{y}) = (\mathbf{x} - \mathbf{y})^T S^{-1} (\mathbf{x} - \mathbf{y}) \quad (3)$$

When feature mismatch has a fixed cost, one frequently uses the Hamming distance:

$$D_{Hamming}(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^n |x_i \oplus y_i| \quad (4)$$

where $(x \oplus y)$ is zero if $x = y$, and 1 otherwise. For binary vectors, this is equivalent to the XOR operation. For two finite point sets, the Hausdorff distance is proposed:

$$D_{Hausdorff}(X, Y) = \max(\sup_{\mathbf{x} \in X} \inf_{\mathbf{y} \in Y} D(\mathbf{x}, \mathbf{y}), \sup_{\mathbf{y} \in Y} \inf_{\mathbf{x} \in X} D(\mathbf{x}, \mathbf{y})) \quad (5)$$

where \sup (*supremum*) denotes the least upper bound, and \inf (*infimum*) denotes the greatest lower bound, respectively. This measure is used for instance in 3D face recognition, where each face may be represented as a point set.

Another important distance measure is based on correlation:

$$D_{Correlation}(\mathbf{x}, \mathbf{y}) = -\frac{n \sum_{i=1}^n x_i y_i - \sum_{i=1}^n x_i \sum_{i=1}^n y_i}{\sqrt{(n \sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2)(n \sum_{i=1}^n y_i^2 - (\sum_{i=1}^n y_i)^2)}} \quad (6)$$

The similarity function can also be replaced with a generative model, where the output takes on a probabilistic interpretation, and the threshold of acceptance is defined on the posterior probability for the biometric sample under the model associated with the claimed identity:

$$s(\mathbf{B}, t_d) = P(M_d | \mathbf{B}) = \frac{p(\mathbf{B} | M_d)p(M_d)}{p(\mathbf{B} | M_d)p(M_d) + p(\mathbf{B} | I)p(I)} \quad (7)$$

Here, M_d denotes the generative model for person d and used in place of the template t_d , I denotes a model for the impostor distribution, and $p(M_d)$ denotes the prior probability of having a genuine claim for person d . The probabilistic formulation is particularly suitable for biometric fusion approaches, where multiple biometric samples are treated as evidence, and the Bayes rule is again used for combining the modalities. We will deal with fusion of multiple biometric modalities in a dedicated section.

An alternative approach of authentication under a client and a generic impostor model is to use the likelihood ratio test:

$$s(\mathbf{B}, M_d, I) = \frac{p(\mathbf{B} | M_d)}{p(\mathbf{B} | I)} \quad (8)$$

where the impostor distribution under infinitely many classes can be replaced with $P(\mathbf{B})$. In (Bazen & Veldhuis, 2004), the authors demonstrate that this measure is more accurate than the posterior probability approach or the distance-based approach with a Euclidean metric for a fingerprint-based authentication application.

Often, the accuracy of the verification system is shown with a *receiver operator characteristic* (ROC)

or a *detection error trade-off* (DET) curve, where FRR is plotted as a function of FAR for a range of τ . Other accuracy indicators are the *equal error rate* (EER), which is the error rate at the point where FAR equals FRR in the DET curve, the *half-total error rate* (HTER), which is the average of FAR and FRR at a given threshold, the *genuine acceptance rate* (GAR), which is equal to $1 - \text{FRR}$, and thus only takes into account the genuine claims, and the *weighted error rate* (WER) which is a weighted version of the EER. The weight R determines how much a false accept is harmful with respect to a false reject:

$$\text{WER}(R) = \frac{P_{\text{FR}} + RP_{\text{FA}}}{1 + R} \quad (9)$$

where P_{FR} and P_{FA} denote the probabilities of false reject and false accept, respectively.

For an identification system, the gallery templates can be ranked according to their similarity to the query. The accuracy of an identification system is often indicated by the *cumulative match characteristic* (CMC) curve, which plots the average rank of the correct gallery template in response to the query. The *rank- n recognition rate* is the correct identification percentage if we assume that a rank below (and including) n is acceptable for identification. The rank system makes sense if a human or a second (and possibly computationally more intensive) system will evaluate the top n candidates for a match with the probe. For the final identification system, the *rank-1 recognition rate* is the most important accuracy measure.

The Structure of a Biometrics System

The information flow in a biometrics system depends on the biometric modality, as well as the operation requirements of the system. For most biometric systems, the operation of the system has the following steps:

1. Acquisition: The acquisition of the biometric is the most important step in the system design. A clean signal that matches the gallery conditions well greatly facilitates recognition. For this reason, biometric evaluations usually test cases where the gallery is acquired under controlled environmental conditions with high quality sensors, and matching is performed under sub-optimal conditions, possibly with inferior sensors.
2. Pre-processing: The biometric data is prepared for the next stage of processing. Under pre-processing, we usually understand cleaning of noise artifacts, normalization, cropping, and signal enhancement operations. In voice biometrics, this step involves segmentation of the speech/non-speech signals. For iris biometrics, the pre-processing deals with illumination changes, contrast normalization, elimination of reflections, defocus and occlusion handling.
3. Registration and segmentation: The biometric template stored in the gallery and the recorded signal must be aligned for obtaining the best results in matching. For face authentication, this is usually performed by using facial landmarks (Tistarelli et al., 2008, Gökberk et al., in press). For iris biometrics, the iris area is segmented at this stage. For modalities with dynamic length, the registration and classification may be performed jointly. We inspect these methods in a separate section.

4. Feature selection and extraction: The raw biometric signal is rarely used for the purposes of classification. The feature extraction method depends on the modality, and influences the subsequent classification stage. For instance in fingerprint-based authentication, the structural features of the fingerprint (i.e. *minutiae*) are extracted at this stage. For iris biometrics, typical features are log-Gabor wavelet coefficients (Daugman, 1988), whereas for voice-based authentication Mel frequency cepstral coefficients are used (Huang et al., 2001).
5. Classification: The classifier decides whether the biometric signal B is generated by the model M or not. Depending on the problem being one of authentication or identification, the classification step assesses the biometric signal against a single model or multiple models. This step may also be followed by a score-based fusion step, if multiple biometric traits are recorded.

Challenges of Biometrics

There are several challenges faced by biometric systems. We mention some of these issues, as they are indicative of future research directions.

Processing and communication load: There is an obvious need for computationally efficient algorithms in biometrics, for several reasons. Some biometrics approaches require great computational resources. For instance in 3D face recognition, the data acquired from the sensor can be massive. The second reason is the real-time requirement imposed by application settings. It is now possible to match a query against millions of fingerprints in a second, but this eludes other modalities at the moment. There is also the issue for algorithms that assess the class-specific variability for a particular modality. For instance the newly acquired Multi-PIE face dataset (Gross et al., 2008) has more than 750.000 samples at 3072 x 2048 resolution (i.e. 6.3 million-dimensional samples), which makes the job of any learning algorithm very difficult.

Template protection: Security and privacy of biometric systems depend on the protection of stored biometric information. Once a biometric is compromised, it is irrevocable, as it is unique to a person. Studies are conducted on *biometric encryption*, where the biometric is combined with a random number to produce a secure and non-reversible hash for authentication.

Cancellable biometrics: For improved privacy, it is possible to store a transformed version of a biometric template. Each application will be associated with a different one-way transformation, and *function creep*, i.e. the use of stored template for other purposes, will be prevented. This kind of a requirement imposes extra constraints on the biometric matching algorithm.

Aging: The aging of the individual causes differences in the stored template and the acquired biometric signal. An aging face has more lines, and it can be difficult to recognize a person from a photograph taken 20 or 30 years ago. It is difficult to collect datasets that span long temporal acquisition periods. Therefore, artificial aging methods are applied to existing data collections to train systems that recognize an aging set of subjects.

Convenience vs. security: The parameterization of a biometric system may allow for a range of custom settings that trade off user convenience and security. For instance, biometrically enhanced entry system of an entertainment park offers high convenience by setting a very low threshold for user admittance, whereas the same system with a high threshold is usable in a high-security nuclear plant.

Score normalization and fusion: Multiple biometric modalities can be used in a single system, which requires efficient and robust score normalization and information fusion algorithms. The best fusion algorithms take into account the correlation structure of the related modalities.

LEARNING AND MATCHING THE BIOMETRIC TEMPLATE

The learning and matching of the biometric template is an integrated problem that encompasses feature selection/extraction and classification, as these steps are invariably intertwined. One may think that once an appropriate pre-processing and feature extraction method is selected, the choice of the classifier is a minor issue. This is not completely true, as literature is replete with studies that contrast various classification approaches for a fixed set of features with significantly different results. Here we give illustrative examples from face recognition domain, and refer the reader to (Kung et al., 2005) and (Petrovska-Delacrétaz et al., 2008) for many applications of learning algorithms for different biometric modalities and their fusion. Our exposition is restricted to two subspace methods for feature extraction to reduce dimensionality, the application of unsupervised clustering to facial landmarking, and combination of weak classifiers for face detection. A later section on dynamic information will deal with Bayesian approaches.

Classification with Subspace Methods

The subspace-based methods are best exemplified with applications to face recognition, which is a high-dimensional problem. The most frequently used baseline method for face recognition is the *Eigenface* method, introduced by Turk & Pentland (1991). This method is based on learning a subspace for faces, where the projected query and target images will be compared. Given a set of training images $\{T_1, \dots, T_k\}$, the covariance matrix C that indicates the distribution of variance is computed:

$$C = \frac{1}{k} \sum_{i=1}^k (T_i - \mu)(T_i - \mu)^T \quad (10)$$

where μ denotes the average face. The eigenvectors of C with the greatest associated eigenvalues (i.e. the *principal components*) denote axes of maximal covariance, and a projection to these axes maximally spreads the projected points. In terms of reconstruction from the projected subspace, the principal component analysis (PCA) method is optimal. If the dimensionality of each image is assumed to be d , the covariance matrix has a dimensionality of $(d \times d)$, usually with $d \gg k$. For this reason, a *singular value decomposition* based method is used to determine at most k eigenvectors for projection. Once the face images are projected to the subspace, several distance measures (e.g. L1, Euclidean) can be used for computing the similarity of the query and the target with nearest-neighbour classification. One measure that has received a lot of interest is the *Mahalanobis cosine distance*. If the p -dimensional subspace projected query is denoted by $u = [u_1, u_2, \dots, u_p]$, and the subspace-projected gallery template is denoted by $v = [v_1, v_2, \dots, v_p]$, denote their corresponding vectors in the Mahalanobis space with unit variance along each dimension as:

$$m_i = \frac{u_i}{\sigma_i} \quad n_i = \frac{v_i}{\sigma_i} \quad (11)$$

where σ_i is the standard deviation for the i^{th} dimension of the p -dimensional eigenspace. Then the Mahalanobis cosine distance is given by:

$$d_{MC}(\mathbf{u}, \mathbf{v}) = \cos(\theta_{mn}) = \frac{\mathbf{m}\mathbf{n}}{|\mathbf{m}||\mathbf{n}|} \quad (12)$$

with θ_{mn} denoting the angle between vectors \mathbf{m} and \mathbf{n} (Ramanathan et al., 2004).

The PCA projection is not optimized for discrimination of patterns. For this reason, *linear discriminant analysis (LDA)* is proposed as an alternative method that seeks the most discriminative subspace projection. Specifically, for c classes denote the grand mean of the training images with $\boldsymbol{\mu}$ and the class means with $\boldsymbol{\mu}_i$. Let the between-class scatter matrix be defined as:

$$S_B = \sum_{i=1}^c k_i (\mathbb{1}_{T_i} - \mathbb{1}_T) (\mathbb{1}_{T_i} - \mathbb{1}_T)^T \quad (13)$$

where k_i denotes the number of training samples for class i . Similarly, the within-class scatter matrix is defined as:

$$S_W = \sum_{i=1}^c \sum_{\mathbf{B}_j \in T_i} (\mathbf{B}_j - \boldsymbol{\mu}_i)(\mathbf{B}_j - \boldsymbol{\mu}_i)^T \quad (14)$$

with T_i denoting training samples of class i . Assuming that S_W is non-singular, the optimal orthonormal projection W_{opt} is selected as the one that maximizes the ratio of the determinant of S_B to the determinant of S_W :

$$W_{opt} = \arg \max_W \frac{|W^T S_B W|}{|W^T S_W W|} = [\mathbf{w}_1 \quad \mathbf{w}_2 \quad \dots \quad \mathbf{w}_m] \quad (15)$$

where $\{\mathbf{w}_i | i=1,2,\dots, m\}$ denotes the set of generalized eigenvectors of S_B and S_W corresponding to the m largest generalized eigenvalues λ_i :

$$S_B \mathbf{w}_i = \lambda_i S_W \mathbf{w}_i \quad (16)$$

The *Fisherface* method (Belhumeur et al., 1997) proposes an alternative criterion to equation (15) to overcome difficulties arising from singular S_W . In the proposed method, the images are projected to a PCA space first. This subspace has a sufficiently low dimensionality (equal to c , the number of classes) to make S_W non-singular. The classification, as before, uses a nearest neighbour approach.

Classifier Combination for Face Detection

Face detection is the essential pre-processing step in face biometrics. This application can be treated as a two-class learning problem, where the positive samples contain face images, and negative samples do not. Osuna et al. (1997) have successfully applied *support vector machines* (SVM) to this problem, and obtained good results. However, practical applications require very fast face detection, for which the SVM-based method was not adequate.

The combination of *weak classifiers* has given rise to one of the most frequently used algorithms in face detection, namely the Viola-Jones algorithm (Viola & Jones, 2001). The core of the method uses the AdaBoost algorithm proposed by Freund and Schapire (1997). In AdaBoost, a sequence of N labelled examples $\langle (\mathbf{x}_1, y_1), \dots, (\mathbf{x}_N, y_N) \rangle$ is used (where \mathbf{x}_i denote the samples, and y_i are binary labels) for finding a good combination of weak learners. The algorithm sets weights w_i for each sample, initialized by a prior distribution that can be uniform in the case no prior information exists. At every iteration of the algorithm, the normalized weights p_i are computed to form a distribution over samples, and the weak learner is called with this distribution to obtain a hypothesis $h_t: X \rightarrow [0,1]$. The error of this hypothesis is given by:

$$\varepsilon_t = \sum_{i=1}^N p_i^t |h_t(\mathbf{x}_i) - y_i| \quad (17)$$

The algorithm sets a weight update parameter $\beta_t = \varepsilon_t / (1 - \varepsilon_t)$. The new weights are set as:

$$w_i^{t+1} = w_i^t \beta_t^{1-|h_t(\mathbf{x}_i)-y_i|} \quad (18)$$

After T iterations, the output of the algorithm is the hypothesis:

$$h_f(\mathbf{x}) = \begin{cases} 1 & \text{if } \sum_{t=1}^T (\log 1 / \beta_t) h_t(\mathbf{x}) \geq \frac{1}{2} \sum_{t=1}^T \log 1 / \beta_t \\ 0 & \text{otherwise} \end{cases} \quad (19)$$

Viola and Jones use the AdaBoost algorithm to select features computed on sub-windows of the image that resemble Haar-wavelets computed in multiple scales (Viola & Jones, 2001). Once computed over the entire image, there are 180.000 possible features available for selection at each stage. In the Viola-Jones algorithm many of these features have final weights set to zero; only 6.000 features are selected. These features are learned in a cascade structure with 38 levels to reduce the processing load of the algorithm. Thus, an average of 10 feature evaluations per sub-window is reported in their experimental study. For the training, 4916 face images (plus their mirror images) and 9544 non-face images (for a total of 350 million sub-windows) were used. This method is currently used as the state-of-the-art benchmark in systems involving face detection¹.

Unsupervised Learning for Feature Localization

Exact localization of facial features in faces ensures better *registration*, subsequently allowing better classification of face images. These prominent features that guide registration are called *anchor points* or *landmarks*. The learning problem for landmarks has two aspects:

- 1) Learning the appearance of each landmark.
- 2) Learning the structural relationship between the landmarks.

It is possible to learn the appearance of each landmark via unsupervised models. For this purpose, features are extracted from around the landmark location from all images in the training set, for each landmark j . Then, a model G_j is fit to the feature distribution. In (Hamouz et al., 2005), complex Gabor features are extracted, and a Gaussian mixture model (GMM) is used for learning the feature distribution. The structural information is taken into account at a later stage. For all the candidates of one landmark (i.e. locations with likelihood exceeding a threshold), two other landmark locations are estimated from the location ground-truth given in the training set, and candidates selected for evaluation. This produces a certain number of landmark triplets for assessment. An SVM classifier is used to produce the final decision.

The GMM (including *k-means clustering* as a special case) is the most frequently used method of clustering in biometrics. The fitting of the GMM involves two decisions: the number of components, and the covariance restrictions imposed on the components. Assuming a particular covariance shape will simplify the observation model, and reduce the number of parameters. For an observation feature of size d , the Gaussian distribution has d parameters for the mean vector. The number of free parameters for the covariance can be as low as one, if a spherical covariance is adapted with the shape of $\sigma^2 I$. Setting σ to unity will result in the *k-means* model. A diagonal covariance will have d parameters, and a full covariance $d(d-1)/2$ parameters. For a mixture model, the number of parameters is multiplied by K , the number of mixture components, plus $(K-1)$ for the component priors. Each simplification corresponds to a different assumption in the model space, and may have adverse effects on the learned model. A diagonal covariance matrix will correspond to assuming independence of observed features, whereas a spherical covariance will also imply that the noise model is the same for each feature dimension.

It should be noted that in modelling a very complex distribution, a large number of simpler Gaussian distributions may be a better solution than fewer full-covariance distributions, particularly due to computational reasons. For instance in the MIT Lincoln Laboratory's successful speaker verification system, a *universal background model* with 2048 diagonal-covariance Gaussian components was employed (Reynolds et al., 2000). An alternative approach for reducing dimensionality is given in (Salah & Alpaydin, 2004), where the mixture components are expressed as *factor analysis* models, which assume that the data are generated in p_i -dimensional manifolds. In this case, each component indexed with i has $d(p_i+1)$ parameters for expressing the covariance, which means that the complexity scale between diagonal- and full-covariance models can be fully explored.

The number of components in a Gaussian mixture can be determined by adapting a criterion of model complexity. In the method proposed by Figueiredo & Jain (2002) a large number of components are fit to the given data set, using the *expectation-maximization* (EM) algorithm. Then, small components are eliminated one by one, each time running EM to convergence. When a single component remains, all converged models are evaluated with a *minimum description length* (MDL) criterion, and one model is

selected. The shape of the covariance must be manually selected at the onset of the algorithm. This is the approach used in (Hamouz et al., 2005) for modelling landmark features.

In (Salah & Alpaydin, 2004), the *incremental mixtures of factor analysers* (IMoFA) algorithm was proposed, which automatically determines the number of components, as well as the covariance complexity for each component separately. In this approach, a single-component factor analyser with a single factor is used to initialize the algorithm. At each iteration, the mixture is grown by adding a component or by adding a factor to an existing component. The algorithm terminates when the likelihood is no longer increasing for a separately monitored validation set.

The IMoFA method was used on Gabor features extracted from sub-windows of face images for landmark localization in (Salah et al., 2007). However, only the best candidate for each landmark was evaluated in the subsequent structural analysis step. The proposed algorithm for this stage (the GOLLUM algorithm) selects one landmark triplet for affine normalization, and models the rest of the landmarks after transformation with Gaussian distributions. The landmark candidates are checked against their expected locations by a thresholded Mahalanobis distance. The landmarks failing the check can be automatically corrected by the back-projection of their expected locations.

DYNAMIC INFORMATION

Biometric information can be recovered from traits with a temporal dimension. A typical example would be the gait of a person, which is unobtrusive and has high user-acceptability, thus capable of serving in a user-convenience scenario. While gait is inherently dynamic, it may be possible to recover temporal information from traits that are usually processed in a static way. For instance the signature or the handwriting of a person can produce dynamic information with appropriate equipment. Similarly, the face is ordinarily a static biometric modality, but sequences of faces in a video stream or the facial expression changes of a person can serve as a dynamic biometrics (Chen et al., 2001).

What makes dynamic biometrics particularly challenging is the variability in the size of the feature vectors. To access the toolbox of methods that work with fixed-dimensionality vectors (e.g. neural networks, PCA, nearest neighbour methods, etc.) *dynamic time warping* may be applied (Kruskal and Liberman, 1983). This method aligns two sequences S_i and S_j by associating cost functions with insertions, deletions, and substitutions, and by locally minimizing the Levenshtein distance, i.e. the minimum number of operations needed to transform S_i into S_j . For feature vectors that have straightforward interpolation functions, resizing the feature vector with linear and quadratic functions to a predetermined size is a computationally cheap alternative. For face recognition, this is the most frequently employed method. For signatures recognition, a re-sampling procedure that produces a fixed-length feature can be employed.

The most frequently used approach in dealing with variable-sized feature vectors is to associate probabilities with feature dynamics. *Dynamic Bayesian networks* (DBNs) are a broad class of graphical models that are capable of incorporating temporal dynamics in this manner. Most frequently used DBNs are *Kalman filters*, and *hidden Markov models* (HMMs). A good survey of learning with DBNs can be found in (Ghahramani, 1998).

It is also possible to extract dynamic information from an otherwise static source by introducing an interest operator, and simulating a dynamic information extraction process. In (Salah et al., 2002), several interest operators are used jointly to derive a saliency map of a facial image, followed by the simulation

of saccadic movements of the fovea in the human visual system to visit the most interesting locations of the image in the order of decreasing saliency. The content of the foveal window is processed via local neural network experts, whereas the location information is mapped to the states of an observable Markov model. After each saccade, Markov models of each person in the gallery are consulted to produce a conditional probability, and if the probability of a particular class exceeds a certain threshold, the identification is effected. With this method, it is possible to inspect only a small part of the facial image before taking a decision, and the whole image is analysed only for difficult cases. Thus, the length of the final feature vector is determined on the fly, and its variability is a blessing, rather than a curse, as it serves reducing the temporal complexity of the identification process. In (Bicego et al., 2003) overlapping windows are extracted from face images, scanned in a regular fashion, and the wavelet coefficients computed from each window are then classified using HMMs.

Storing multiple templates for a single user can make a biometrics system more robust, at the cost of increased computational complexity. For dynamic biometrics, it may be a necessity to store multiple templates. For instance in signature recognition, a person may have an elaborate, slow-dynamics signature, and a fast-dynamics, quick and dirty signature for rapid signing. For this purpose, it will be necessary to quantify the similarity of biometric signals under different dynamic models. Given two different biometric signals \mathbf{B}_i and \mathbf{B}_j , and two DBNs M_i and M_j trained for responding maximally to these signals (or to a set of signals containing them), a similarity (or affinity) score can be computed as:

$$s(\mathbf{B}_i, \mathbf{B}_j) = \frac{1}{2} \left\{ \frac{1}{\tau_j} \log P(\mathbf{B}_j | M_i) + \frac{1}{\tau_i} \log P(\mathbf{B}_i | M_j) \right\} \quad (20)$$

where τ_i is the authentication threshold for model M_i . Panuccio et al. proposed another measure of similarity (2002), which also takes into account the model quality, i.e. how well M_i models the signal B_i :

$$s(\mathbf{B}_i, \mathbf{B}_j) = \frac{1}{2} \left\{ \frac{P(\mathbf{B}_i | M_j) - P(\mathbf{B}_i | M_i)}{P(\mathbf{B}_i | M_i)} + \frac{P(\mathbf{B}_j | M_i) - P(\mathbf{B}_j | M_j)}{P(\mathbf{B}_j | M_j)} \right\} \quad (21)$$

In both approaches, $P(\mathbf{B}_j | M_i)$ denotes the probability of observing \mathbf{B}_j under the model M_i . Once the similarity matrix is computed, it is possible to employ clustering methods to group the signals into several clusters of differing dynamics, and train one model per cluster for the final system.

For a typical biometrics application, an obvious problem with these approaches is that limited enrollment time requirement leaves the system with very few examples, whereas the dimensionality can be large. One way of dealing with the dimensionality is to constrain the models appropriately, effectively reducing the number of parameters in the process. If the dynamics have a well-defined order, as in a signature, where the order of letters do not change, some transitions may be pruned from the model, leaving less parameters to estimate. An example is using left-to-right HMMs instead of fully connected HMMs, where each state represents a dynamic that is executed once, before continuing to the next dynamic. Depending on the particular case, the time spent in one state may be short or long. Another method is to constrain the observation distribution at every state of the model. Typically Gaussian models or Gaussian mixtures are used for observation probability distribution, where the complexity can be tuned by imposing restrictions on the covariance.

The temporal dynamics can also be taken into account by constant online modification of static models, instead of a single dynamic model that traces the complete evolution of the dynamic. This is particularly useful for dynamics with indeterminate duration, for instance in camera-based surveillance applications. An illustrative example is video-based biometrics, where the motion of bodies or faces is analysed. This necessitates reliable foreground-background segmentation, which is frequently tackled by learning statistical models for foreground objects and/or the background image. For static views, Stauffer and Grimson have proposed an influential method, where one adaptive Gaussian mixture model per background pixel is trained (1999). In their approach, the recent history of each background pixel X_t is modelled by a mixture of K Gaussian distributions, and the probability of a particular pixel belonging to the background at any time is given by:

$$P(X_t) = \sum_{i=1}^K \pi_i N(X_t, \mu_i, \sigma_i^2 I) \quad (22)$$

where π_i is the component prior, μ_i is the mean, and σ_i is the standard deviation that specifies the spherical covariance shape of the component. Pixels not falling within 2.5 standard deviations from the mean of any component are assumed to be evidence for new components, and replace the component with the smallest prior. At any time, the first few distributions that account for a fixed portion of the data are considered to belong to the background.

MULTIPLE BIOMETRICS AND INFORMATION FUSION

Information fusion in biometrics serves a twofold purpose. First and foremost, the system requirements may dictate an operational description beyond the technological provisions of a single biometric modality, either in terms of security, or user convenience. Multiple biometrics can be used to design systems to fit more demanding requirements. The second purpose of using multiple modalities relates to user-convenience. It is known that some biometric modalities (like fingerprints) are not usable for a small but non-negligible percentage of the population (Newham, 1995), and consequently, a non-discriminating biometric authentication scheme needs to accommodate these users.

Assume two biometric signals \mathbf{B}_1 and \mathbf{B}_2 are recorded from a single person b in two different modalities with models $M_1 = \{s_1, \mathbf{t}_d^1, \tau_1\}$ and $M_2 = \{s_2, \mathbf{t}_d^2, \tau_2\}$, respectively, where s_i denotes the similarity function, \mathbf{t}_d^i is the biometric template and τ_i is the threshold of authentication. The combined biometric system can be made more secure than either of the individual biometric systems by requiring:

$$s_1(\mathbf{B}_1, \mathbf{t}_b^1) > \tau_1 \wedge s_2(\mathbf{B}_2, \mathbf{t}_b^2) > \tau_2 \quad (23)$$

or it can provide for more robust (e.g. for cases where the data acquisition conditions are impaired for one modality), or more convenient (e.g. letting the user decide which modality to use) systems by allowing alternative authentication venues, at the cost of reduced system security:

$$s_1(\mathbf{B}_1, \mathbf{t}_b^1) > \tau_1 \vee s_2(\mathbf{B}_2, \mathbf{t}_b^2) > \tau_2 \quad (24)$$

This is the simplest biometric fusion scenario at the decision level of individual systems. It is possible to effect fusion of biometric information at the raw data level, at the feature level, at the matching score level, or at the decision level. Another dimension of fusion is the architecture, which can be serial or parallel. In (Gökberk et al., 2005), the authors contrast a serial (hierarchical) fusion scheme for 3D face recognition in which the first classifier ranks the most plausible classes, after which the second classifier operates on a reduced set of classes, with parallel fusion schemes in which all classifier outputs are jointly assessed. The parallel approach has increased real-time operation cost, but its accuracy is superior to that of the serial, and both fusion approaches excel in comparison to individual classifiers.

When using multiple systems, we usually deal with scores that have different ranges and meanings. The system proposed in (Gökberk et al., 2005) sidesteps this issue by operating on rank scores, but the appropriate normalization of scores is nonetheless an important issue. The normalization can be performed for each score domain separately, bringing the scores to a common scale for combination (*transformation-based score fusion*). Weighted combination of scores is a simplified version of this approach, and most of the fusion approaches reported in the literature aim at learning an appropriate set of weights for individual biometric modalities. For the authentication task, this can also be achieved by treating the genuine and impostor scores as two separate classes, and applying supervised classification methods. For instance in (Ross & Jain, 2003), the scores from different modalities (face, fingerprint, hand geometry) are combined with:

1. The SUM rule, which corresponds to a weighted average of the scores, where learning involves finding the best set of weights on the training set,
2. A decision tree, which was used to learn a number of rules to solve the two-class classification problem of distinguishing impostor and genuine claims,
3. A linear discriminant classifier, which is obtained by projecting the scores to a subspace that minimizes the within-class variance and maximizes the between-class variance, for the same two-class problem.

The SUM rule has been found to perform better than the alternatives, and has the additional benefit of having a threshold to calibrate the system for increased security or convenience. In this kind of a *classifier-based fusion scheme*, the distribution of genuine and impostor scores is usually not balanced. A zero-cost impostor model implies that the samples available for impostor access are $N-1$ times greater than the number of genuine access samples, N being the number of subjects, and assuming equal number of samples per subject.

An additional difficulty is that most classification approaches would not allow the tuning of the system for a particular FAR or FRR requirement. Suppose the system is required to operate at 0.001 FAR, and a neural network classifier is selected for solving the genuine-impostor classification problem. The training regime needs to be adjusted appropriately, and most likely the final classifier output needs to be post-processed (e.g. by applying a threshold on the posterior) to bring the final system into the desired operation range.

In (Nandakumar et al., 2008) the genuine and impostor match scores from different modalities are modelled with Gaussian mixtures, and the decision follows a likelihood ratio test. This is a *density-based score fusion* scheme, as opposed to transformation-based or classifier-based fusion schemes. The difficulty in density-based fusion is the fitting of the model; it is not uncommon that the limited number of samples available for statistical learning rules out all but the simplest models. However, if the underly-

ing joint score density is known, it is possible to formulate an optimal fusion scheme. The application of the likelihood ratio test is similar to the case in equation (8). We summarize this method here, along with further justification of its use.

Assume $\mathbf{B} = [B_1, B_2, \dots, B_k]$ denotes the biometric match scores from k different modalities. Assume further that we know the joint densities of the genuine claims and impostor claims, denoted with $p(\mathbf{B}|M_d)$ and $p(\mathbf{B}|I)$. Let Ψ denote a statistical test for the null-hypothesis H_0 : \mathbf{B} is associated with a claim from an impostor versus H_1 : \mathbf{B} is associated with a genuine claim. $\Psi(\mathbf{B})$ is the binary valued function for accepting or rejecting the null hypothesis. The justification proposed by (Nandakumar et al., 2008) for using the likelihood-ratio test rests on the Neyman-Pearson theorem, which states that for testing H_0 against H_1 there exists a test Ψ and a constant η such that:

$$P(\Psi(\mathbf{B}) = 1|H_0) = \alpha \quad (25)$$

and

$$\Psi(\mathbf{B}) = \begin{cases} 1, & \text{when } \frac{p(\mathbf{B}|M_d)}{p(\mathbf{B}|I)} \geq \eta, \\ 0, & \text{when } \frac{p(\mathbf{B}|M_d)}{p(\mathbf{B}|I)} < \eta. \end{cases} \quad (26)$$

The more important part of the theorem states that if a test Ψ satisfies these equations for some constant η , then it is the *most powerful test* for testing H_0 against H_1 at level α . Consequently, the authors propose to employ the ratio of $p(\mathbf{B}|M_d)$ to $p(\mathbf{B}|I)$ for giving the authentication decision, and η is selected in a way that the likelihood ratio test maximizes the GAR at the specified FAR. In practice, the true genuine and impostor score densities are unknown, and they are approximated with the help of the training set, typically with mixture models. One obvious difficulty is that due to user convenience issues, there are very few (typically less than five) genuine scores per subject.

The likelihood-ratio test can further be refined by learning subject-specific parameters for the threshold η or for the class-conditional densities (Poh & Kittler, 2007). The impostor model can be a single model (an example of which is the universal background model frequently used in speaker verification) or it can be a set of alternative models, in which case it is called a *cohort*. The latter approach is less popular, as it implies increased computational complexity.

A discriminative biometric modality will result in a good separation of the genuine and impostor score distributions, and produce higher ratios (i.e. a more confident decision). The fusion should take into account the discriminative power of the contributing biometrics, but it can also consider an assessment of the acquisition conditions and the *quality* of the recorded biometric signal, which can be different for each instance, depending on the acquisition conditions and the particular sensor used. Once a set of quality measures are selected, a *quality-based fusion* approach can be adopted. In (Nandakumar et al., 2008), the quality measures are taken into account by replacing the genuine and impostor score distributions with joint distributions of scores and corresponding quality measures. In (Chatzis et al., 1999) a vector quantization approach was taken to define score categories based on quality, and the biometric

scores are fuzzified to give a robust decision. In (Maurer & Baker, 2007) Bayesian belief networks were employed to incorporate the quality as an effect on the prior.

A recent evaluation campaign under the European FP6 Network of Excellence BIOSECURE assessed biometric fusion methods under *cost* and *quality* considerations (Poh et al., in press). Here, cost refers to the computational cost of verification, which is important for real-time operation. Scores obtained from fingerprint and face scanners were associated with a cost of use, and with several automatically derived quality measures. 22 score-level fusion systems were evaluated in the campaign, including dynamic fusion approaches where biometric modalities were consulted in a sequential fashion until a desired level of authentication confidence was reached or all the scores were exhausted.

Biometric information can be fused with ancillary information that has little discriminative value, but nonetheless can increase the robustness of the system in question. By making the assumption that these so-called *soft biometric traits* (e.g. height, gender) are independent, they can be fused with a strong biometric:

$$s(\mathbf{B}, \mathbf{t}_b) = a_0 \log P(\mathbf{t}_b^0 | \mathbf{B}_0) + \sum_{i=1}^m a_i \log P(\mathbf{B}_i | \mathbf{t}_b^i) \quad (27)$$

where \mathbf{B}_0 is the primary biometric with a large weight a_0 , and \mathbf{B}_i are the soft biometrics with relative weights a_i (Jain et al., 2004).

The usefulness of fusion increases if the individual modalities that are fused are not highly correlated. For this reason, it is important to understand the role of the multimodal databases when analysing fusion results. Collecting biometric information can be a meticulous process, and the anonymity of the subjects is imperative. Since collecting and associating multiple biometrics from individuals is perceived to be a threat to anonymity, some of the studies in the literature employ *chimeric databases*, which are created by randomly combining biometric information from different modalities under virtual identities. Since the correlation structure is not retained, the added value of fusion will be overestimated under a chimeric database.

To illustrate this point, consider a system with face and palm-print modalities. In a natural database, male faces will be associated with larger palm-prints and the face and palm skin colours will be highly correlated, whereas a chimeric dataset will lose these correlations. One potential solution is to employ a small set of natural multimodal samples and use unsupervised clustering to model the covariance structure. Then recorded biometrics can independently be assigned to the nearest cluster, and used to construct realistic chimeric datasets by probabilistic selection of samples from within clusters. Some biometric combinations like faces and fingerprint have low correlation, and thus more suitable for chimeric databases.

EVALUATING A BIOMETRICS SYSTEM

The practice of biometrics often requires the evaluation of a system under scrutiny to determine the operating region, as well as the comparison of several algorithms in terms of their computational cost, accuracy, and robustness. Several important considerations that are common practice in the machine learning community should guide the biometrics researcher in the evaluation of the developed systems.

This section provides a short list of good practices. For a detailed exposition, see (Mansfield & Wayman, 2002).

Appropriate experimental setup: The model development and reporting of the test results should be performed on different datasets. When reporting results, accuracies on the training set and the test set should be reported in conjunction, as the amount of difference in accuracy is indicative of the generalization capability of a model. While developing the models, the test set should not be consulted at all, as doing so would imply that the test set is contributing to the learning process, and the accuracies must be appropriately adjusted. A separate validation set can be set aside from the training set during model selection and learning.

Statistical tests for comparison: The claim of superiority of one algorithm over another cannot be based on a higher accuracy alone; it should be supported with statistical tests. Especially with benchmark databases, the accuracies reported in the literature tend to become very high over time, and it is important to demonstrate the success of new algorithms over alternative approaches with tests administered under exactly the same protocol. Producing mean and standard deviation of accuracy results with randomized runs allows comparisons in terms of distance in standard deviations, which serves as a heuristic to decide the significance of obtained accuracy differences. The reader is referred to (Yildiz & Alpaydin, 2006) for statistical testing methodology to determine the algorithm with the smallest expected error given a data set and a number of learning algorithms.

Cross-database evaluation: The problem of overlearning plagues many learning algorithms. Considering that most biometrics databases are constituted by samples acquired under similar conditions (i.e. same environment, same sensor, similar pre-processing and compression, etc.) it is important to assess the generalization ability of proposed algorithms by cross-session recordings, where training and testing are performed on data recorded in different sessions with temporal separation. Better still are the cross-database evaluations, where the training and testing are performed with different databases. Sometimes a *world model* is used to learn the conditions specific to the particular data acquisition environment. This is a small and representative data sample acquired under the test conditions that is used in addition to the training set.

Challenges and evaluation campaigns: The independent testing of the biometrics systems has been a vital practice. The efforts of the U.S. National Institute of Standards and Technology (NIST) have been instrumental and exemplary in this respect, and a number of campaigns have been conducted by NIST and other institutions to evaluate biometric systems independently (see for example, Philips et al. 2007). Among these, we should mention NIST's Face Recognition Vendor Test, speaker recognition challenges, Iris Challenge Evaluation, Face Recognition Grand Challenge and the recent Multiple Biometrics Grand Challenge. Apart from NIST, important biometric assessment campaigns are NVU iris challenges, Univ. of Bologna Fingerprint Verification Competition, M2VTS and BANCA challenges, and the BioSecure Multimodal Evaluation Campaign. These are valuable venues to try baseline algorithms with clearly defined protocols, to see the shortcomings of present approaches, and to objectively evaluate the performance of a new system. The reader is referred to (Petrovska-Delacrétaz et al., 2008) and to references therein for more information on biometric evaluation campaigns.

Links to databases and codes: Table 1 includes some links for the most important databases for different biometric modalities. For OpenSource code of baseline systems the reader is referred to the web resources of Association BioSecure², which includes OpenSource reference systems,

Table 1. Links to some important biometrics databases.

Name	Modality	Link
BANCA	face video	www.ee.surrey.ac.uk/CVSSP/banca
CASIA	iris & gait palmprint	www.cbsr.ia.ac.cn
FERET	2D face	www.itl.nist.gov/iad/humanid/feret/feret_master.html
FRGC	2D-3D face	face.nist.gov/frgc
FVC	fingerprint	atvs.ii.uam.es/databases.jsp
MCYT	signature & fingerprint	atvs.ii.uam.es/databases.jsp
NIST	speech	www.ldc.upenn.edu (multiple corpora available from Linguistic Data Consortium)
PolyU	palm-print	www.comp.polyu.edu.hk/biometrics
USF	gait	figment.csee.usf.edu/GaitBaseline (includes code)
XM2VTS	face & speech	www.ee.surrey.ac.uk/CVSSP/xm2vtsdb

publicly available databases, assessment protocols and benchmarking results for several modalities (2D and 3D face, speech, signature, fingerprint, hand, iris, and talking-face).

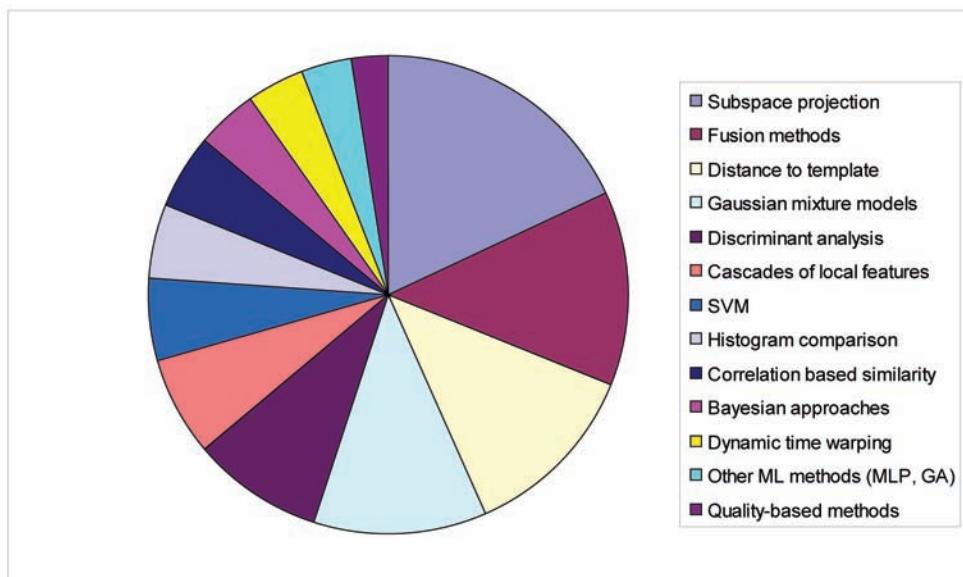
Applications: Figure 1 shows the distribution of various ML approaches in biometrics, as gleaned from the 125 papers presented at the International Conference on Biometrics (Lee & Li, 2007). The most frequently used methods are based on learning a suitable subspace projection to reduce the dimensionality. Non-parametric methods that store templates for each enrolled subject and look at the distance to the query from the template are employed in more than 10 per cent of the papers, with different distance measures. Subspace projections favour Mahalanobis-based distance functions, whereas Hamming distance and histogram comparison techniques are frequently applied in iris recognition. Gaussian mixture models are particularly prominent in speech applications; HMM and other dynamic Bayesian approaches are used in signature and speech applications. About 15 per cent of all the papers favoured a fusion approach.

CONCLUSION

The rapid advance in technology, particularly the production of cheaper and better sensors enables computer systems to automatically identify people. This capability satisfies an increasing need for security and smarter applications, and gains wide acceptance thanks to strict control for privacy and ethical concerns.

In this chapter, we have given a glimpse of machine learning methods that are relevant in making biometric technology a reality. These methods are actively used in deployed systems, but there is ever the need for faster and more accurate algorithms. We hope that the application examples and references we have provided will serve the reader in creating novel machine learning solutions to challenging biometrics problems.

Figure 1. Distribution of different ML approaches to biometric problems.



ACKNOWLEDGMENT

This research is supported by the Dutch BRICKS/BSIK project. The author thanks Dr. Eric Pauwels and an anonymous reviewer for valuable comments.

REFERENCES

- Bazen, A. M., & Veldhuis, R. N. J. (2004). Likelihood-ratio-based biometric verification. *IEEE Trans. Circuits and Systems for Video Technology*, 14(1), 86–94. doi:10.1109/TCSVT.2003.818356
- Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19, 711–720. doi:10.1109/34.598228
- Bicego, M., Castellani, U., & Murino, V. (2003). Using hidden Markov models and wavelets for face recognition. In *Proc. of the Int. Conf. Image Analysis and Processing* (p. 52).
- Bolle, R. M., Connell, J. H., Pankanti, S., Ratha, N. K., & Senior, A. W. (2004). *Guide to biometrics*. New York: Springer-Verlag.
- Chatzis, V., Bors, A. G., & Pitas, I. (1999). Multimodal decision-level fusion for person authentication. *IEEE Trans. System, Man, and Cybernetics, part A*, 29(6), 674-680.
- Chen, L. F., Liao, H. Y. M., & Lin, J. C. (2001). Person identification using facial motion. In *Proc. of the Int. Conf. Image Processing* (pp. 677-680).

Daugman, J. (1988). Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36, 1169–1179. doi:10.1109/29.1644

Figueiredo, M. A. T., & Jain, A. K. (2002). Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24, 381–396. doi:10.1109/34.990138

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1), 119–139. doi:10.1006/jcss.1997.1504

Ghahramani, Z. (1998). Learning dynamic Bayesian networks. In C. L. Giles & M. Gori (Eds.), *Adaptive processing of sequences and data structures* (pp. 168–197). Berlin, Germany: Springer-Verlag.

Gökberk, B., Salah, A. A., & Akarun, L. (2005). Rank-based decision fusion for 3D shape-based face recognition. In *Proc. of the Int. Conf. Audio- and Video-based Biometric Person Authentication* (LNCS 3546, pp. 1019–1028).

Gökberk, B., Salah, A. A., Alyüz, N., & Akarun, L. (in press). 3D face recognition: Technology and applications. In M. Tistarelli, S. Z. Li, & R. Chellappa (Eds.), *Biometrics for surveillance and security*. London: Springer-Verlag.

Gross, R., Matthews, I., Cohn, J., Kanade, T., & Baker, S. (2008). Multi-PIE. In *Proc. of the 8th IEEE Conf. on Automatic Face and Gesture Recognition*, Amsterdam.

Hamouz, M., Kittler, J., Kamarainen, J. K., Paalanen, P., Kalviainen, H., & Matas, J. (2005). Feature-based affine-invariant localization of faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(9), 1490–1495. doi:10.1109/TPAMI.2005.179

Huang, X., Acero, A., & Hon, H. (2001). *Spoken language processing*. Upper Saddle River, NJ: Prentice Hall.

Jain, A. K., Dass, S. C., & Nandakumar, K. (2004). Soft biometric traits for personal recognition systems. In *Proc. of the Int. Conf. Biometric Authentication*, Hong-Kong.

Kruskal, J., & Liberman, M. (1983). *The symmetric time-warping problem: From continuous to discrete*. Reading, MA: Addison-Wesley.

Kung, S. Y., Mak, M. W., & Lin, S. H. (2005). *Biometric authentication: A machine learning approach*. Upper Saddle River, NJ: Prentice Hall Professional Technical Reference.

Lee, S.-W., & Li, S. Z. (2007). *Advances in biometrics* (LNCS 4642). Berlin, Germany: Springer Verlag.

Mansfield, A. J., & Wayman, J. L. (2002). *Best practices in testing and reporting performance of biometric devices*. Centre for Mathematics and Scientific Computing, National Physical Laboratory.

Maurer, D. E., & Baker, J. P. (2007). Fusing multimodal biometrics with quality estimates via a Bayesian belief network. *Pattern Recognition*, 41(3), 821–832. doi:10.1016/j.patcog.2007.08.008

- Nandakumar, K., Chen, Y., Dass, S. C., & Jain, A. K. (2008). Likelihood ratio-based biometric score fusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2), 342–347. doi:10.1109/TPAMI.2007.70796
- Newham, E. (1995). *The biometrics report*. SJB Services.
- Osuna, E., Freund, R., & Girosi, F. (1997). Training support vector machines: An application to face detection. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 130-136).
- Panuccio, A., Bicego, M., & Murino, V. (2002). A hidden Markov model-based approach to sequential data clustering. In *Structural, syntactic and statistical pattern recognition* (LNCS 2396, pp. 734-742). Berlin, Germany: Springer-Verlag.
- Petrovska-Delacrétaz, D., Chollet, G., & Dorizzi, B. (Eds.). (2008). *Guide to biometric reference systems and performance evaluation*. London: Springer-Verlag.
- Phillips, P.J., Scruggs, W. T., O'Toole, A. J., Flynn, P. J., Bowyer, K. W., Schott, C. L., & Sharpe, M. (2007). *FRVT 2006 and ICE 2006 large-scale results*. NISTIR 7408.
- Poh, N., Bourlai, T., Kittler, J., Allano, L., Alonso, F., Ambekar, O., et al. (in press). Benchmarking quality-dependent and cost-sensitive multimodal biometric fusion algorithms. *IEEE Trans. Information Forensics and Security*.
- Poh, N., & Kittler, J. (2007). On the use of log-likelihood ratio based model-specific score normalisation in biometric authentication. In S.-W. Lee & S. Z. Li (Eds.), *Advances in biometrics: Proc. of the ICB* (LNCS 4642, pp. 614-624). Berlin, Germany: Springer-Verlag.
- Ramanathan, N., Chellappa, R., & Chowdhury, A. K. R. (2004). Facial similarity across age, disguise, illumination, and pose. In *Proc. of the Int. Conf. Image Processing* (pp. 1999-2002).
- Reynolds, D., Quatieri, T., & Dunn, R. (2000). Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10, 19–41. doi:10.1006/dspr.1999.0361
- Ross, A., & Jain, A. (2003). Information fusion in biometrics. *Pattern Recognition Letters*, 24, 2115–2125. doi:10.1016/S0167-8655(03)00079-5
- Salah, A. A., & Alpaydın, E. (2004). Incremental mixtures of factor analysers. In *Proc. of the Int. Conf. on Pattern Recognition* (pp. 276-279).
- Salah, A. A., Alpaydın, E., & Akarun, L. (2002). A selective attention-based method for visual pattern recognition with application to handwritten digit recognition and face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3), 420–425. doi:10.1109/34.990146
- Salah, A. A., Çınar, H., Akarun, L., & Sankur, B. (2007). Robust facial landmarking for registration. *Annales des Télécommunications*, 62(1-2), 1608–1633.
- Stauffer, C., & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 246-252).

Sun, Z., Wang, Y., Tan, T., & Cui, J. (2005). Improving iris recognition accuracy via cascaded classifiers. *IEEE Trans. Systems, Man, and Cybernetics. Part C: Applications and Reviews*, 35(3), 435–441. doi:10.1109/TSMCC.2005.848169

Tistarelli, M., Bicego, M., Alba-Castro, J. L., Gonzàlez-Jimènez, D., Mellakh, A., Salah, A. A., et al. (2008). 2D face recognition. In D. Petrovska-Delacrétaz, G. Chollet, & B. Dorizzi (Eds.), *Guide to biometric reference systems and performance evaluation* (pp. 217-266). London: Springer-Verlag.

Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1), 71–86. doi:10.1162/jocn.1991.3.1.71

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Xudong, J., & Ser, W. (2002). Online fingerprint template improvement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8), 1121–1126. doi:10.1109/TPAMI.2002.1023807

Yildiz, O., & Alpaydin, E. (2006). Ordering and finding the best of $K>2$ supervised learning algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3), 392–402. doi:10.1109/TPAMI.2006.61

Zhang, D., Wai-Kin, K., You, J., & Wong, M. (2003). Online palmprint identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9), 1041–1050. doi:10.1109/TPAMI.2003.1227981

KEY TERMS AND DEFINITIONS

biometric: A biometric is a personal or a behavioural trait that can be used to identify a person.

Biometric Template: The biometric template of a person is a pre-recorded biometric sample stored in a database for later authentication.

Authentication: Authentication is the decision process where a biometric is sampled from a person with an identity claim, and the sampled biometric is compared to a biometric template stored previously for this person to validate the identity claim.

Biometric System: A biometric system involves a set of sensors to record a biometric from the users of the system, a database of stored biometric templates, and an authentication algorithm by which the recorded biometric is compared to the template.

Biometric Fusion: The use of multiple biometric samples in a biometrics system. These samples can be collected through different modalities, resulting in multimodal fusion, or multiple samples from a single modality or even a single sensor can be employed for fusion.

ENDNOTES

¹ The code is available in the OpenCV library, at <http://sourceforge.net/projects/opencvlibrary>

² <http://biosecure.it-sudparis.eu/AB/>

Chapter 27

Neural Networks for Modeling the Contact Foot–Shoe Upper

M. J. Rupérez

Universitat Politècnica de València, Spain

J. D. Martín

Universitat de València, Spain

C. Monserrat

Universitat Politècnica de València, Spain

M. Alcañiz

Universitat Politècnica de València, Spain

ABSTRACT

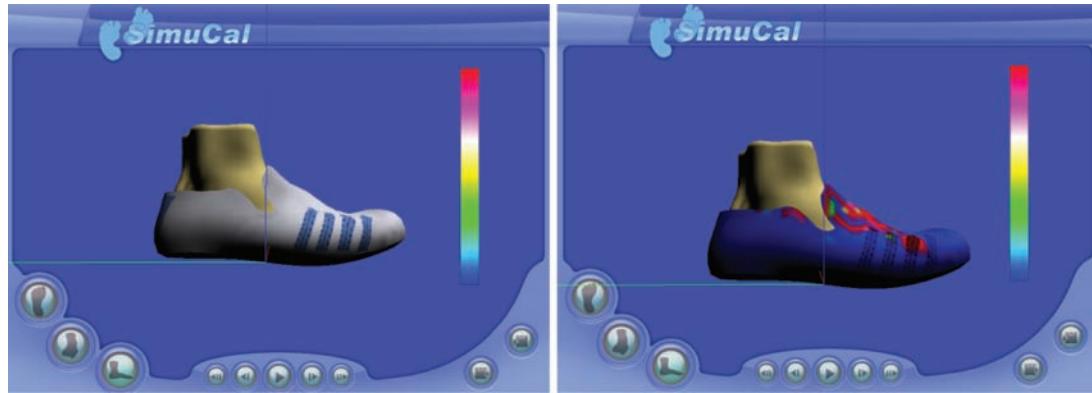
Recently, important advances in virtual reality have made possible real improvements in computer aided design, CAD. These advances are being applied to all the fields and they have reached to the footwear design. The majority of the interaction foot–shoe simulation processes have been focused on the interaction between the foot and the sole. However, few efforts have been made in order to simulate the interaction between the shoe upper and the foot surface. To simulate this interaction, flexibility tests (characterization of the relationship between exerted force and displacement) are carried out to evaluate the materials used for the shoe upper. This chapter shows a procedure based on artificial neural networks (ANNs) to reduce the number of flexibility tests that are needed for a comfortable shoe design. Using the elastic parameters of the material as inputs to the ANN, it is possible to find a neural model that provides a unique equation for the relationship between force and displacement instead of a different characteristic curve for each material. Achieved results show the suitability of the proposed approach.

INTRODUCTION

Consumers are demanding higher levels of footwear comfort and functionality and the footwear designers are aware of these aspects when a new collection is designed. Two tests are currently used to assess the footwear comfort and functionality: Subjective tests based on user perceptions with feet standard

DOI: 10.4018/978-1-60566-766-9.ch027

Figure 1. SIMUCAL, Footwear Virtual Simulator



in size and form and objective tests based on the measure of biomechanical variables during the use of footwear in both real and simulated conditions. The use of the virtual reality and simulation techniques provide a viable alternative to these procedures which are costly and time consuming (Figure 1).

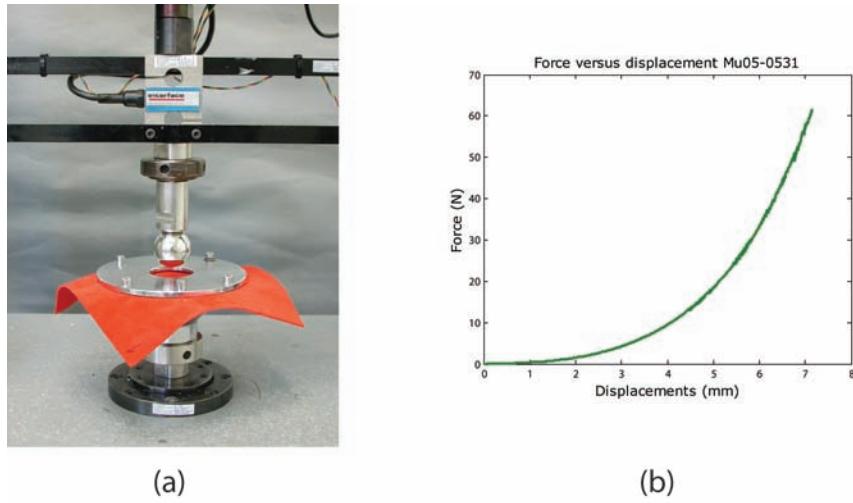
Comfort is very important in footwear manufacture, because the foot is one of the structures of the human body that supports most weight in gait. It is also the principal shock absorber of the impact against the ground. Most of the biomechanical research related to the foot has been focused on measuring the plantar pressure distribution, since the presence of high pressures at this zone is one of the major causes of diabetic ulcerations of the foot and pain in the soft tissues (Holewski et al., 1989; Onwuanyi, 2000). However, the shoe uppers also take part in comfort (Jordan & Bartlett, 1995) and their deformation can be the cause of pain and injury on the foot surface.

The deformation in the shoe upper material is due mainly to the contact with the foot surface. In order to model this contact, a whole study about the elasticity of the shoe upper material must be performed. Several tests must be done to determine the elastic parameters that characterize it. In this chapter, a model to characterize shoe upper materials is proposed. The model is aimed at reducing the quantity of elasticity tests necessary to determine the behavior of the materials subjected to this kind of contact.

As the foot surface has little soft tissue, foot surface behavior can be assumed to be completely determined by the behavior of the bones that form it, and in this sense, a set of tests called *flexibility tests* were designed for the Institute of Biomechanics of Valencia. These tests consist of applying a controlled force by means of a sphere that simulates a bone, on a sample of material fixed to a gag so that the slide was avoided (Figure 2a). The result of these tests is the measure of the force that the sphere exerts on the sample versus the displacement of the central point of the material shell (Figure 2b).

Three materials that are commonly used in the footwear manufacture were subjected to these tests in order to simulate how the material was deformed by pressure of the bones. With the results of these tests that provide the curves of the force exerted versus the displacement of the central point of the material shell, it is possible to draw an expression that characterizes the material. Different materials are usually separately modeled. However, there are several approaches within Machine Learning (ML) that allow obtaining a unique equation for all the materials. In this chapter, we show how a Multilayer Perceptron (MLP) is able to characterize all three materials with a unique equation that has the following inputs: the elastic parameters of the corresponding material (Young modulus and Poisson coefficient), the thickness

Figure 2. (a) Test of the flexibility for the upper. (b) Force versus material displacement.



and the displacement of the central point of the material shell. The output of the network is given by the force that the sphere exerts for that displacement.

The three materials used were different kinds of leather. The relevance of obtaining a unique equation for the three materials stems from the fact that this equation will model the behavior under contact of the most of leathers used in footwear manufacture, just by introducing their elastic parameters and the thickness. It should be emphasized that this procedure can be applied to model the behavior under contact of other materials like for example, foams used as reinforcements of the ankle and heel, and hence, the contact with the foot surface of any kind of shoe upper could be modeled. ANNs involve a novel approach with respect to the classical techniques of modeling each material separately.

It should be pointed out that ML in general, and Artificial Neural Networks (ANNs) in particular, have been widely applied in many different fields but their use in Biomechanics is still scarce. In Biomechanics, ANNs have been applied for characterization of the human movement and for biomechanical modeling of the movement control, but they have not been used to characterize any material. Here, ANNs also involve a novel approach with respect to the classical techniques of modeling.

LITERATURE REVIEW

Biomechanics: Foot-Shoe Interaction

As it has been mentioned before, most of the biomechanical research related to the foot has been focused on measuring the plantar pressure distribution. To measure these pressures, foot models (Chen et al., 2001; Cheung et al., 2005) and shoe models have been developed; however the shoe models that are available only simulate the behavior of the soles and the behavior of the materials they are made of. In (Lemmon et al., 1997), insoles are modeled as hyper-elastic foams by using a 2D finite element model of plane strain. The planar pressure was measured in the second metatarsal in order to study the effect

of insoles thickness. In (Chen et al., 2001), the planar pressure was measured in the entire foot-sole interface with a 3D finite element analysis. In this study, soles were modeled with different combinations of hyper-elastic materials. In (Verdejo & Mills, 2004), midsoles made of EVA (Ethylene-vinyl acetate) foam were modeled to analyze their durability.

There are not many models to simulate the contact between foot and shoe upper. To the authors' knowledge, the only model that simulates the deformation of materials in shoe uppers in gait was proposed by (Rupérez et al., 2008), in this work the authors presented the basis of a computer application which could be an alternative to the subjective procedures currently being used in footwear comfort tests (García-Hernández et al., 2005). In these tests, prototypes are made and then evaluated by users; these tests are costly and time-consumer. With the computer application presented by these authors, the manufacturer is provided with the capability to value functional features such as the fit and the foot-shoe interaction for the different kinds of shoes designed, using only the computer. Our approach becomes especially relevant within this framework, a fast characterization of the materials would allow the footwear designer or manufacturer to test a new design with more materials and in a more realistic way, for example adding the reinforcements to the upper, doing faster the analysis and reducing the time of the design process.

Artificial Neural Networks

ANNs have been widely and successfully applied to many different fields. In particular, its application to Medicine is extensive and relatively usual (Lisboa, 2002). Although clinical decision support systems have used Artificial Intelligence (AI) methods since the end of the fifties (Ledley & Lusted, 1959), it was only during the nineties that decision support systems were routinely used in clinical practice on a significant scale. A reference work in this field is found in (Brier et al., 1995), in which the capabilities of ANNs and NONMEN are benchmarked. Predictions of steady state peak and trough serum gentamicin concentrations were compared using a traditional population kinetic method and the computer program NONMEM, to an empirical approach using neural networks. Peak serum concentration predictions using neural networks were statistically less biased and showed comparable precision with paired NONMEM predictions. The prediction errors were lower for the neural networks than for NONMEM, and the neural network reproduced the observed multimodal distribution more accurately than NONMEM. It was concluded that neural networks can predict serum drug concentrations of gentamicin, and thus, may be useful in predicting the clinical pharmacokinetics of drugs.

Predicting the clinical pharmacokinetics of drugs has been one of the main fields of ANNs application. There are a number of applications related to anaesthesia due to the difficult control of its depth, being especially remarkable (Nebot et al., 1996), (Zhang et al., 2002), and (Leistritz et al., 2002). Diabetes is also a deeply studied field of research (Hernando et al., 1996), (Mougiakakou & Nikita, 2000), (Trajanoski et al., 1998). Neural networks have also been applied to the assessment of HIV immunopathology (Hatzakis & Tsoukas, 2001), (Sardari & Sardari, 2002) and to the monitoring of the immunosuppressive drugs that avoid the organ rejection or patient intoxication after kidney transplantation (Martín-Guerrero et al., 2003a) and (Martín-Guerrero et al., 2003b).

Use of ANNs in Biomechanics is not very wide, although ANNs are finding new fields of application within the Clinical Biomechanics since they are able to develop from a simple classification procedure up to a signal processing and analysis. ANNs could become standard methods in Clinical Biomechanics. In Biomechanics, development of ANNs is taking place in a wide range of modalities found in the analysis

of online data; their main applications have been carried out for the classification of people's movements based on several characteristic variables and for biomechanical modeling, where the sequence of input and output variables follow common biomechanical ideas about the movement control, without having deterministic relationships of these variables like EMG (Electromyography)-Force (Schöllhorn, 2004).

In the classification of people's movement, ANNs were used for the classification of gait data, the classification of different movement tasks and for the classification of EMG data.

In 1993, (Holzreiter & Köle, 1993) classified gait pathologies by means of ground forces, Fast Fourier transforms (FFTs) of vertical forces served as inputs to a standard supervised MLP. With adequate training of the MLPs, discrimination of healthy and pathological gait was achieved an accuracy of up to 95%. In 1997, (Bishop et al. 1997) investigated several forms of MLP in assigning sensed low back pain during a set of trunk motions to selected kinematic parameters that were collected with a triaxial goniometer. ANN classifier produced superior results with up to 85% accuracy on test data. Regarding the classification of the EMG data, neuromuscular disorders constitute a significant cause of disability in young and old people, a group from the Cyprus Institute of Neurology and Genetics investigated the application of several ANNs in the field of EMG analysis extraordinary systematic (Schöllhorn, 2004).

In biomechanical modeling, ANNs are particularly attractive for Clinical Biomechanics in order to model the elusive relationship between EMG, kinetic, and kinematic parameters, because of their highly non-linear mapping characteristics and many researchers attempted to apply feed forward MLPs to cope with several facets of this aspect (Schöllhorn, 2004).

There are applications in many other fields, different from Medicine and Clinical Biomechanics. Finances, for example, is fruitful field of application of ANNs (Wong & Selvi, 1998), being especially remarkable the attempts of predicting sales and the evolution of the stock market. Another fields are Industry (Widrow et al., 1994), Marketing (Curry & Moutinho, 1993), Communications (Plumbley, 1997) or Animal Science (Fernández et al., 2006), only to mention a few of them.

In summary, practical applications of ANNs have become enormous in the last twenty years, and basically, they have been applied to many different practical problems in which one of the following tasks must be carried out: classification, regression, clustering or time series prediction. However, although they have been used in analysis gait, as far as the authors know there is not any research on the study of the applicability of ANNs for the characterization of the contact between shoe upper and foot surface, thus being the current work a novel approach in this field.

METHODS AND RESULTS

Materials and Tests

Three materials that are commonly used in the manufacture of shoe uppers were provided by a footwear manufacturer. Four samples of each material were subjected to tensile tests according to the UNE EN ISO 527-1 norm. In order to obtain their elastic parameters, the stress versus the longitudinal strain and versus the constriction were measured. A linear fitting in the stress-strain curves was made for the small strain zone, thereby obtaining the elastic modulus and the Poisson ratio of each material (Table 1).

To evaluate the upper elasticity, the IBV (Institute of Biomechanics of Valencia) carried out tests that allowed determining the stiffness of the upper material simulating the pressure exerted by the foot. The tests consisted of applying a controlled force by means of a sphere of diameter 3.6 cm on a sample

Table 1. Thickness, elastic modulus, and Poisson ratio of each material

Material	Thickness (mm)	E (MPa)	Poisson ratio
I	0.65	8.489	0.061
II	0.76	12.904	0.015
III	0.77	5.783	0.056

of material fixed to a gag so that the slide is avoided (Figure 2a).

Samples of the upper materials were subjected to these tests. A gag of 40mm of interior diameter was used. The tests were quasi-static, the velocity was 0.08mm/s and as limit of the force 60N was taken. The results are shown in the Figure 3. These graphs represent the force that the sphere exerted versus the displacement of the central point of the shell of material.

Artificial Neural Networks

The use of ANNs is proposed in order to obtain a general model for the simulation of the contact foot-shoe upper that allows characterizing the force versus displacement for different materials at the same time. This way, it would no longer be necessary the individual characterization of each material, thus reducing the number of tests to carry out. An additional advantage is that with a general model, it would also possible to obtain the characterization for new materials using the same model, albeit results should be understood with caution if the new material considered is considerably different to those used to obtain the neural model.

An MLP has been used for modeling since it is a universal function approximator with generalization capabilities which is especially interesting in real problems such as that tackled in this work. An MLP is composed of a layered arrangement of artificial neurons in which each neuron of a given layer feeds all the neurons of the next layer. The first layer contains the input nodes, which are usually fully-connected to hidden neurons and these, in turn, to the output layer (Haykin, 2008). A single neuron and the traditional model of the multilayer neural network are shown in Fig. 4 (left and right, respectively).

The inputs x_i to the neuron are multiplied by adaptive coefficients w_i called weights, which represent the synaptic connectivity between neurons. The output of a neuron is usually taken to be a sigmoid-shaped (sigmoid or hyperbolic tangent function) φ :

Figure 3. Force versus Displacement

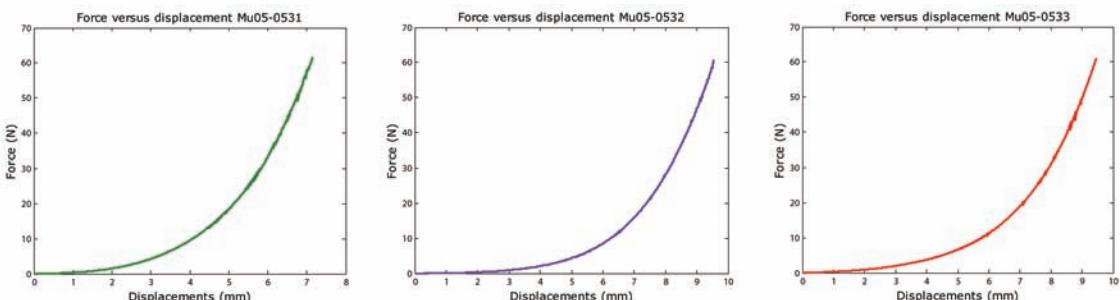
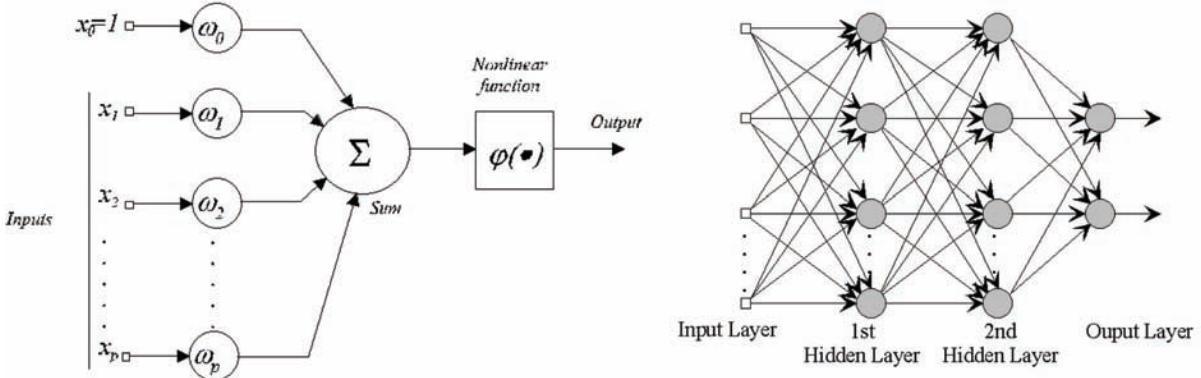


Figure 4. Left: Elementary neuron model. Right: feedforward network, where each line represents a synaptic connection (no feedback connections exist).



$$output_j = \varphi\left(\sum_{i=0}^P w_{ij} \cdot x_i\right) \quad (1)$$

In Eq. (1), the subindex j represents the j -th neuron, P the number of input patterns and $x_0=1$ is the bias input to the neuron. The structure is completely defined by taking x_i to be the external inputs, and $output_j$ to be the output of the neuron.

The network was trained using the Expanded Range Algorithm (ERA) with a momentum term to speed up convergence. ERA is a method to mitigate the effect of local minima in gradient-descent algorithms (Gorse et al., 1997). In this technique, the training set for the network is expanded progressively from an initial average setting to the original complete training set. The expansion process generates different sets of training data and is controlled by an homotopy parameter $\lambda \in [0, 1]$. Initially, $\lambda=0$ y $d_p(0) = \langle d \rangle$, where $\langle d \rangle$ denotes the arithmetic mean of the original p example training patterns and $d_p(0)$ is the initial setting of the training set in the ERA algorithm. This training set is then expanded for each training epoch, in a step-wise manner, until $\lambda=1$, at which point the training set is identical to the original set of input/output pairs. If, for simplicity, we assume a structure with a single output neuron, the expansion of the training set as a function of λ , $d_p(\lambda)$ is given by Eq. (2):

$$d_p(\lambda) = \langle d \rangle + \lambda [d_p - \langle d \rangle] \quad (2)$$

where d_p is a member of the original training set.

The ability of the ERA algorithm to avoid local minima of the error function is based on three assumptions:

1. The problem defined by $\lambda=0$ has only one global minimum.
2. The first small step ($\lambda=\eta < 1$) away from the $\lambda=0$ solution keeps the system in a global minimum.
3. The range can be progressively expanded to $\lambda=1$ without displacing the system from the global minimum at any step.

Table 2. Results achieved in the MLP modeling of the exerted force to the sphere. All values are expressed in Newton.

	Training data set	Test data set
ME	0.001	0.002
MAE	0.055	0.054
RMSE	0.110	0.107

These three points are satisfied for the XOR problem (Gorse et al., 1997); however, ERA can break down in more complex situations. The reason for this is that no general strategy has been found for step size variation during the course of the expansion of the training set so as to ensure global convergence. This is a difficult problem due to the dependence on the training set, the network architecture and the initialization of the synaptic weights. In this work, λ was increased from 0 to 1 in a homogenous step-wise manner. In particular, ten steps were considered, and thus, if t is the number of training iterations, λ value will increase 0.1 each $t/10$ iterations.

The following features were used as inputs to the MLP model: displacement (mm), thickness (mm), Young's modulus (N/m) and Poisson's ratio. The output of the model is the exerted force hence the MLP is aimed at modeling the exerted force depending on the displacement and the analyzed material characteristics.

Experimental data was split into two data sets: a training data to obtain the neural network model and a test data to test the model performance with different data from those used to obtain the model. This way, generalization capabilities are guaranteed. The training data set was formed by 6543 patterns whereas the test data set consisted of 3272 patterns. Training was early stopped by cross-validation.

The model architecture and parameters were optimized by trial-and-error procedures. The best neural model showed an architecture $4 \times 6 \times 1$. Synaptic weights were initialized following a pseudo-random normal distribution with mean zero and standard deviation unity. The learning rate was set to 0.01 and the momentum constant was set to 0.5. The optimum instant to stop the training procedure was after 2977 iterations.

Table 2 shows a summary of the achieved results. The following error functions are given:

1. *Mean Error (ME)*. It is a measure of the prediction bias:

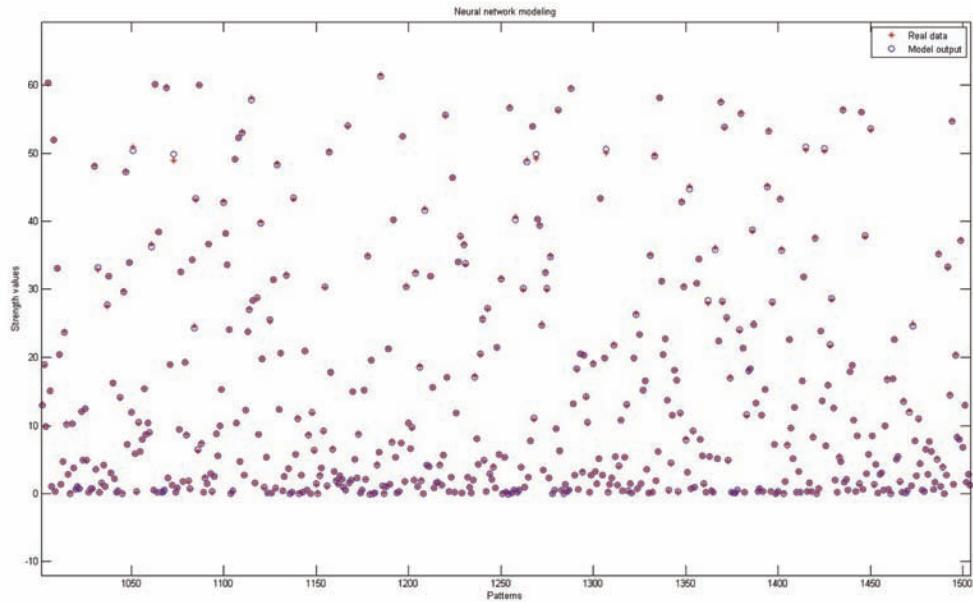
$$ME = \frac{1}{N} \sum_{i=1}^N d_i - o_i \quad (3)$$

where d_i is the desired outcome for the i -th pattern, o_i is the predicted value and N is the number of patterns used to compute the ME.

2. *Mean Absolute Error (MAE)*. It is a measure of prediction accuracy. It is an intuitive measure since it provides a sort of mean value of the committed errors:

$$MAE = \frac{1}{N} \sum_{i=1}^N |d_i - o_i| \quad (4)$$

Figure 5. Neural network modeling (circles) and real values of force (stars) for approximately half of the patterns belonging to the validation data set (random selection of patterns).



3. *Root-Mean Square Error (RMSE)*. It is also a measure of prediction accuracy but more statistically robust:

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (d_i - o_i)^2} \quad (5)$$

In order to assess the errors provided in Table 2, it is necessary to know the range of desired values. In particular, in our database force values were ranged from 0 to $61.71N$, being the mean value equal to $12.86N$, and the standard deviation $16.02N$. Therefore, on average, relative errors are lower than 1%, which is an accurate regression, as it is shown in Figure 5, where part of the patterns belonging to the validation data set together with the real values are plotted. Plotted patterns have been randomly selected to provide a useful representation, since the performance for all the patterns was approximately similar.

CONCLUSION AND FUTURE WORK

In this work, we have presented a real application of ANNs to Biomechanics. In particular, the goal is modeling the exerted force on different materials used for shoe manufacture as a function of the displacement and the material characteristics in the simulation of the contact between the foot and the shoe upper. The final aim is to have a general model that is able to simulate the exerted force for different materials. This way, the quantity of flexibility tests needed to characterize each material would be

drastically reduced since usually flexibility tests are carried out for each material separately.

Achieved results have shown that an MLP-based modeling has a high level of accuracy, and with only one model, it has been possible to model three different materials. Therefore, it is possible to use a unique mathematical function that given the material characteristics and the corresponding displacement, models the exerted force with high accuracy.

Our ongoing research is focused on developing the simulation of the contact between the foot and the shoe upper in a virtual way, so a fast characterization of the used materials is very useful to reduce the computational cost. Future work will be devoted to get the whole simulation of the shoe deformation in gait, including both upper and soles, so we will apply this procedure to characterize the rest of footwear components like reinforcements, insoles etc., in order to improve the development of a virtual tool that allow footwear manufacturers to analyze their designs in a much faster way.

REFERENCES

- Bishop, J. B., Szalpski, M., Ananthraman, S., McIntrye, D. R., & Pope, M. H. (1997). Classification of low back pain from dynamic motion characteristics using an artificial neural network. *Spine*, 22(24), 2991–2998. doi:10.1097/00007632-199712150-00024
- Brier, M. E., Zurada, J. M., & Aronoff, G. R. (1995). Neural network predicted peak and trough gentamicin concentrations. *Pharmaceutical Research*, 12(3), 406–412. doi:10.1023/A:1016260720218
- Chen, W. P., Tang, F. T., & Ju, C. W. (2001). Stress distribution of the foot during mid-stance to push-off in barefoot gait: A 3-D finite element analysis. *Clinical Biomechanics (Bristol, Avon)*, 16, 614–620. doi:10.1016/S0268-0033(01)00047-X
- Cheung, J. T., Zhang, M., Leung, A. K., & Fan, Y. B. (2005). Three-dimensional finite element analysis of the foot during standing-a material sensitivity study. *Journal of Biomechanics*, 38, 1045–1054. doi:10.1016/j.jbiomech.2004.05.035
- Curry, B., & Moutinho, L. (1993). Neural networks in marketing: Modelling consumer responses to advertising stimuli. *European Journal of Marketing*, 27(7). doi:10.1108/03090569310040325
- Fernández, C., Soria, E., Martín, J. D., & Serrano, A. J. (2006). Neural networks for animal science applications: Two case studies. *Expert Systems with Applications*, 31(2), 444–450. doi:10.1016/j.eswa.2005.09.086
- García-Hernández, J., Heras, S., Alfons, J., Paredes, R., Nácher, B., & Alemany, S. (2005). The morfo3D foot database. *Pattern Recognition and Image Analysis*, 3523, 658–665.
- Gorse, D., Shepherd, A. J., & Taylor, J. G. (1997). The new ERA in supervised learning. *Neural Networks*, 10(2), 343–352. doi:10.1016/S0893-6080(96)00090-1
- Hatzakis, G., & Tsoukas, C. (2001). Neural networks in the assessment of HIV immunopathology. In *Proceedings of the AMIA Symposium* (pp. 249-53).
- Haykin, S. (2008). Neural networks: A comprehensive foundation (3rd ed.). New York: Macmillan.

- Hernando, M. E., Gomez, E. J., del Pozo, F., & Corcoy, R. (1996). DIABNET: A qualitative model-based advisory system for therapy planning in gestational diabetes. *Medical Informatics*, 21(4), 359–374.
- Holewski, J. J., Moss, K. M., Stess, R. M., Graf, P. M., & Grunfeld, C. (1989). Prevalence of foot pathology and lower extremity complications in a diabetic outpatient clinic. *Journal of Rehabilitation Research and Development*, 26, 35–44.
- Holzreiter, S. H., & Köle, M. E. (1993). Assessment of gait patterns using neural networks. *Journal of Biomechanics*, 26(6), 645–651. doi:10.1016/0021-9290(93)90028-D
- Jordan, C., & Bartlett, R. (1995). Pressure distribution and perceived comfort in casual footwear. *Gait & Posture*, 3, 215–220. doi:10.1016/0966-6362(96)82850-5
- Ledley, R. S., & Lusted, L. B. (1959). Reasoning foundations of medical diagnosis. *Science*, 130, 9–21. doi:10.1126/science.130.3366.9
- Leistritz, L., Kochs, E., Galicki, M., & Witte, H. (2002). Prediction of movement following noxious stimulation during 1 minimum alveolar anesthetic concentration isoflurane/nitrous oxide anesthesia by means of middle latency auditory evoked responses. *Clinical Neurophysiology*, 113(6), 930–935. doi:10.1016/S1388-2457(02)00064-0
- Leinson, D., Shiang, T. Y., Hashmi, A., Ulbrecht, J. S., & Cavanagh, P. R. (1997). The effect of insoles in therapeutic footwear - a finite element approach. *Journal of Biomechanics*, 30, 615–620. doi:10.1016/S0021-9290(97)00006-7
- Lisboa, P. J. G. (2002). A review of evidence of health benefit from artificial neural networks in medical intervention. *Neural Networks*, 15(1), 11–39. doi:10.1016/S0893-6080(01)00111-3
- Martín-Guerrero, J. D., Camps-Valls, G., Soria-Olivas, E., Serrano-López, A. J., Pérez-Ruixo, J. J., & Jiménez-Torres, N. V. (2003a). Dosage individualization of erythropoietin using a profile-dependent support vector regression. *IEEE Transactions on Bio-Medical Engineering*, 50(10), 1136–1142. doi:10.1109/TBME.2003.816084
- Martín-Guerrero, J. D., Soria-Olivas, E., Camps-Valls, G., Serrano-López, A. J., Pérez-Ruixo, J. J., & Jiménez-Torres, N. V. (2003b). Use of neural networks for dosage individualisation of erythropoietin in patients with secondary anemia to chronic renal failure. *Computers in Biology and Medicine*, 33(4), 361–373. doi:10.1016/S0010-4825(02)00065-3
- Mougiakakou, S. G., & Nikita, K. S. (2000). A neural network approach for insulin regime and dose adjustment in type 1 diabetes. *Diabetes Technology & Therapeutics*, 2(3), 381–389. doi:10.1089/15209150050194251
- Nebot, A., Cellier, F. E., & Linkens, D. A. (1996). Synthesis of an anaesthetic agent administration system using fuzzy inductive reasoning. *Artificial Intelligence in Medicine*, 8(2), 147–166. doi:10.1016/0933-3657(95)00030-5
- Onwuanyi, O. N. (2000). Calcaneal spurs and plantar heel pad pain. *The Foot*, 10, 182–185. doi:10.1054/foot.2000.0633

Plumbley, M. (1997). Communications and neural networks: Theory and practice. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'97) - Volume 1* (p. 135).

Rupérez, M. J., Monserrat, C., & Alcañiz, M. (2008). Simulation of the deformation of materials in shoe uppers in gait. Force distribution using finite elements. *Int J Interact Des Manuf Biomech*, 2, 59–68. doi:10.1007/s12008-008-0036-6

Sardari, S., & Sardari, D. (2002). Applications of artificial neural network in AIDS research and therapy. *Current Pharmaceutical Design*, 8(8), 659–670. doi:10.2174/1381612024607199

Schöllhorn, W. I. (2004). Applications of artificial neural nets in clinical biomechanics. *Clinical Biomechanics (Bristol, Avon)*, 19, 876–898. doi:10.1016/j.clinbiomech.2004.04.005

Trajanoski, Z., Regitnig, W., & Wach, P. (1998). Simulation studies on neural predictive control of glucose using the subcutaneous route. *Computer Methods and Programs in Biomedicine*, 56(2), 133–139. doi:10.1016/S0169-2607(98)00020-0

Verdejo, R., & Mills, N. J. (2004). Heel-Shoe interactions and the durability of EVA foam running-shoe midsoles. *Journal of Biomechanics*, 37, 1379–1386. doi:10.1016/j.jbiomech.2003.12.022

Widrow, B., Rumelhart, D. E., & Lehr, M. A. (1994). Neural networks: Applications in industry, business and science. *Communications of the ACM*, 37(3), 93–105. doi:10.1145/175247.175257

Wong, B. K., & Selvi, Y. (1998). Neural network applications in finance: A review and analysis of literature (1990-1996). *Information Management*, 34(3), 129–139. doi:10.1016/S0378-7206(98)00050-0

Zhang, X. S., Huang, J. W., & Roy, R. J. (2002). Modeling for neuromonitoring depth of anesthesia. *Critical Reviews in Biomedical Engineering*, 30(1-3), 131–173. doi:10.1615/CritRevBiomedEng.v30.i123.70

KEY TERMS AND DEFINITIONS

Shoe Upper: Material, usually leather, which the top part of the shoe is made of and it normally covers the top surface of the foot.

Elasticity Test: Tests carried out to obtain the elastic parameter of a material.

Flexibility Test: Tests specially designed to simulate how a bone pushes the shoe upper material. These tests consisted of applying a controlled force by means of a sphere that simulated a bone on a sample of material fixed to a gag so that the slide was avoided.

Young's Modulus (E): describes tensile elasticity, or the tendency of an object to deform along an axis when opposing forces are applied along that axis; it is defined as the ratio of tensile stress to tensile strain. It is often referred to simply as the *elastic modulus*.

Poisson's Ratio: (ν): is the ratio of the relative contraction strain, or transverse strain (normal to the applied load), divided by the relative extension strain, or axial strain (in the direction of the applied load).

Electromyography (EMG): is a technique for evaluating and recording the activation signal of muscles. EMG is performed using an instrument called an electromyograph, to produce a record called an electromyogram. An electromyograph detects the electrical potential generated by muscle cells when these cells contract, and also when the cells are at rest.

Global Minimum: It is the minimum value of an error function. In the framework of artificial neural networks, the independent variables that define the error function are usually the synaptic weights.

Local Minima: Local minima of the error function. They do not correspond with the lowest value of the error but only with the lowest error within a limited range of values of the independent variables. Gradient-descent algorithms strongly depend on their initialization in order to avoid local minima.

ERA Algorithm: It stands for Expanded Range Approximation. It is a variant of the classical Back-propagation algorithm. It is aimed at working with “smooth” function errors, in which it is more likely to avoid local minima.

Chapter 28

Evolutionary Multi–Objective Optimization of Autonomous Mobile Robots in Neural–Based Cognition for Behavioural Robustness

Chin Kim On
Universiti Malaysia Sabah, Malaysia

Jason Teo
Universiti Malaysia Sabah, Malaysia

Azali Saudi
Universiti Malaysia Sabah, Malaysia

ABSTRACT

The utilization of a multi-objective approach for evolving artificial neural networks that act as the controllers for phototaxis and radio frequency (RF) localization behaviors of a virtual Khepera robot simulated in a 3D, physics-based environment is discussed in this chapter. It explains the comparison performances among the elitism without archive and elitism with archive used in the evolutionary multi-objective optimization (EMO) algorithm in an evolutionary robotics study. Furthermore, the controllers' moving performances, tracking ability and robustness also have been demonstrated and tested with four different levels of environments. The experimentation results showed the controllers allowed the robots to navigate successfully, hence demonstrating the EMO algorithm can be practically used to automatically generate controllers for phototaxis and RF-localization behaviors, respectively. Understanding the underlying assumptions and theoretical constructs through the utilization of EMO will allow the robotics researchers to better design autonomous robot controllers that require minimal levels of human-designed elements.

DOI: 10.4018/978-1-60566-766-9.ch028

INTRODUCTION

Evolutionary Robotics (ER) is a methodology to develop a suitable control system for autonomous robots with minimal or without human intervention at all through evolutionary computation, to adapt itself to partially unknown or dynamic environments (Floreano, 2000b; Nelson, 2006; Pratihar, 2003). In other words, ER is defined as the synthesis of autonomous robots and/or controllers using artificial evolutionary methods (Teo, 2004a; Teo, 2005). ER is mainly seen as a strategy to develop more complex robot controllers (Floreano, 1996; Floreano, 2000b). Algorithms in ER frequently operate on a population of candidate controllers, initially selected from some random distributions (Alba, 2002; Urzelai, 2000). The evolutionary processes involve a set of operators, namely selection, crossover, mutation and other genetic algorithm (GA) operators (Coello, 2005a; Floreano, 2000a). Using different approaches of evolution, for example Genetic Algorithm, Genetic Programming, Co-evolution, and anytime learning, researchers strive for an algorithm that is able to train robots to perform their tasks without external supervision or help.

A number of studies have already been successfully conducted in evolutionary robotics for phototaxis, phonotaxis and obstacle avoidance tasks (Floreano, 2000; Teo, 2005; Floreano, 1996; Floreano, 2000a; Horchler, 2004; Reeve, 2005). In previous studies related to phototaxis tasks, the researchers used a fixed amount of hidden neurons in the neural network or just a two-layer neural network for their robot's task (Floreano, 1996; Floreano, 2000b; Teo, 2004b). They have not emphasized on the relationship between the robot's behavior and its corresponding hidden neurons. Additionally, to the best of our knowledge, there have not been any studies conducted yet in applying the multi-objective algorithm in evolving the robot controllers for phototaxis behavior. Besides, the literature also showed that other researchers have successfully synthesized some fitness functions to evolve the robots for the required behaviors (Floreano, 1996; Floreano, 2000a; Floreano, 2000b; Nolfi, 2000). However, the fitness functions used can be further improved or augmented in order to increase the robot's ability in completing more complex tasks (Marco, 1996). In addition, research regarding radio frequency (RF) signal localization has yet to be studied in ER. The RF signal is defined as radio frequency signal (abbreviated RF, rf, or r.f.) (Gibson 2007; Pike, 2007; NOAA 2008). It is a term that refers to an alternating current having characteristics such that, if the current is an input to an antenna, an electromagnetic field is generated suitable for wireless broadcasting and/or communications used (Gibson 2007; Pike, 2007; NOAA 2008). The RF signal source has provided the capability for improvements in tracking, search and rescue efforts. As such, robots that are evolved with RF-localization behavior may potentially serve as an ideal SAR assistant (Gibson 2007; Pike, 2007; NOAA 2008).

Previous studies of evolution mainly focus on achieving a single objective and they were unable to explicitly trade-off in terms of their different objectives when there is problem that involves more than one objective (Billard, 2001; Floreano, 2000b). With respect to the other ANN studies, the EMO application is advantageous compared to some conventional algorithms, such as backpropagation, conventional GAs, and Kohonen SOM network (Alba, 2002; Floreano, 1996; Urzelai, 2000). For example, the number of hidden neurons used in multiple layer perceptrons (MLP) and the number of cluster centers in Kohonen's SOM network need to be determined before training (Alba, 2002; Floreano, 1996; Urzelai, 2000). Meanwhile, the traditional learning methods for ANNs such as backpropagation usually suffer from the inability to escape from local minima due to their use of gradient information (Nehmzow, 1992; Teo, 2005). In addition, EMOs are able to solve two or more objectives in a single evolutionary process compared to conventional GAs (Teo, 2005). An EMO emphasizes on the generation of Pareto optimal

set of solutions that explicitly trade-offs between more than one conflicting and different optimization objectives. Commonly, any number of objectives can be optimized with the EMO methodology, as long as the objectives are in conflict with one another. On the other hand, the use of EMO in optimizing non-conflicting objectives would simply result in a convergence to a single solution rather than a set of Pareto-optimal solutions that trade-off in terms of the conflicting objectives. In this research, two conflicting objectives are investigated: (1) maximization of the robot's task behavior, and (2) minimization of the ANN's complexity in terms of the number of hidden nodes. In previous work done by Teo (Teo, 2005), it was proven that the multi-objective evolutionary algorithm was able to discover reasonably good solutions with less computational cost if compared to a single-objective EA, a weight-sum EMO, and a hand-tuned EMO algorithm for simulated legged robot. Thus, it was observed that the application of EMO into ER is one of the alternatives in recent years' research.

In this study, the Pareto Differential Evolution (PDE) algorithm is used as the primary evolutionary optimization algorithm (Abbass, 2001; Teo 2005). There are two behaviors to be optimized; phototaxis and RF-localization behavior respectively. Thus, there are two experiments conducted and discussed in this chapter. Firstly, there are two distinct objectives to be optimized for the phototaxis behavior: (1) maximizing the robot's phototaxis behavior, and (2) minimizing the complexity of the neural controller in terms of the number of hidden units used. There are also two distinct objectives to be optimized for the RF-localization behavior: (1) maximizing the robot's RF-localization behavior whilst (2) minimizing the neural network complexity. Thus, a multi-objective approach (Abbass, 2001; Abbass, 2002; Storn, 1995), which combines the Differential Evolution algorithm and Pareto multi-objective concepts is used, namely the PDE-EMO algorithm. The elitist without archive and elitist with archive PDE-EMO are used to generate a Pareto optimal set of ANNs that acts as the robot's controller. Typically, an elitist algorithm is used to avoid losing good solutions during the EMO optimization process (Coello, 2005). In the study of (Abbass, 2001; Abbass, 2002; Coello, 2005a; Coello, 2005b; Zitzler, 2004; Zitzler, 2005), it was clearly shown that elitism helps in achieving better convergence in EMOs. However, there are still some arguments found among the elitism with archive and elitism without archive performances' comparison (Coello, 2005b; Zitzler, 2004). The reviewed literature have clearly showed that the elitism with archive helps in achieving better convergence in EMOs compared to the elitism without archive. Nevertheless, this study will further shows that the controllers evolved with elitism without archive performed better than the controllers that were evolved using elitism with archive. Thus, this study's main objective is to verify the performance and the ability of the generated robot controllers using the elitism with archive and elitism without archive PDE-EMO algorithm in the ER study, for both phototaxis and RF-localization behaviors. Furthermore, the best Pareto solutions obtained are tested for their robustness in four different levels of maze environments. The testing environments are different from the evolution environment. The complexity of the testing environment used is increased with the number of obstacles involved.

The remainder of this paper is organized as follows. Section II discusses the background of the research area. In sections III and IV, the ANNs and methodology used in evolving the robot controllers are discussed. In section V, a detailed explanation of the experimental setup used for both phototaxis and RF-localization behaviors evolution is given. The maze environments used during testing are also included in section V. Further, the fitness function used is presented in section VI. In section VII and VIII, the algorithms' performance comparisons and simulations results are discussed respectively. Finally, the last section of this chapter summarizes the conclusions and future work arising from this study.

BACKGROUND

A number of researches have been conducted in the ER field (Billard, 2001; Cliff, 1993; Floreano, 1996; Floreano, 2000a; Floreano, 2000b; Horchler, 2004; Marco, 1996; Nehmzow, 1992; Nelson, 2006; Pratihar, 2003; Reeve, 2005; Teo, 2005; Urzelai, 2000). Researchers investigate on designing, through artificial evolution and neural mechanism in order to generate the controllers that needed. Some of the designed controllers are having special capabilities or behaviors to perform the tasks given in the preserved environment, such as phototaxis, phonotaxis, obstacle avoidance, box pushing, exploration among others ((Billard, 2001; Cliff, 1993; Floreano, 1996; Floreano, 2000a; Floreano, 2000b; Horchler, 2004; Marco, 1996; Nehmzow, 1992; Nelson, 2006; Pratihar, 2003; Reeve, 2005; Teo, 2005; Urzelai, 2000)). These authors have successfully synthesized some fitness functions to evolve the robots for the required behaviors. However, the fitness functions used are still can be further improved or augmented in order to increase the robot's ability in completing more complex tasks. Some of the researchers utilized Backpropagation, conventional Genetic Algorithms (GAs), and Kohonen SOM network to generate the required controllers (Billard, 2001; Cliff, 1993; Floreano, 1996; Floreano, 2000a; Floreano, 2000b; Horchler, 2004; Marco, 1996; Nehmzow, 1992; Nelson, 2006; Pratihar, 2003; Reeve, 2005; Teo, 2005; Urzelai, 2000)). These studies have demonstrated the capability of ANNs for the required robot behaviors by using an evolutionary optimization technique. Interestingly however, there are still only a handful of studies in applying EMO as well as RF localization behavior in ER.

The research of EMO had been started at the middle of 1980s (Abbass, 2004; Coello, 2005a; Coello, 2005b, Deb, 2007; Zitzler, 2004; Zitzler, 2005), a term named as Multi-objective Optimization Evolutionary Algorithm (MOEA) was utilized instead of EMO (Coello, 2005a; Zitzler, 2005). Since then, a number of studies had been carried out in that area. The MOEA was mainly used in machine learning study (Coello, 2005a; Zitzler, 2005). Since then, a wide variety of algorithms have been proposed (Abbass, 2004; Deb, 2007; Coello, 2005a; Coello, 2005b, Zitzler, 2004; Zitzler, 2005). Some of the researchers have proposed variations of Schaffer's Vector Evaluation Genetic Algorithm (VEGA) or other similar population-based approaches in order to further improve the existing population selection technique during the optimization processes (Coello, 2005a). Until 1991, Goldberg introduced a new population selection method in his famous book on genetics algorithm (Coello, 2005b, Zitzler, 2004). The proposed population selection method is named as Pareto optimality (Coello, 2005b, Zitzler, 2004). Niching or fitness sharing was suggested by Goldberg as a way to maintain the diversity and avoid convergence of the GA into a single solution (Coello, 2005b, Zitzler, 2004). A number of EMO algorithms have been proposed after Goldberg idea (Coello, 2005b, Zitzler, 2004). Such as Nondominated Sorting Genetic Algorithm (NSGA), Niched-Pareto Genetic Algorithm (NPGA), Multi-Objective Genetic Algorithm (MOGA), Strength Pareto Evolutionary Algorithm (SPEA), Strength Pareto Evolutionary Algorithm 2 (SPEAII), Pareto Archived Evolution Strategy (PAES), Nondominated Sorting Genetic Algorithm II (NSGA-II), Niched Pareto Genetic Algorithm 2 (NPGA 2), Pareto Envelope-based Selection Algorithm (PESA), Micro Genetic Algorithm and others (Coello, 2005b, Zitzler, 2004).

Abbass, (2001) proposed a new concept with hybridized the conventional Pareto-based algorithm with the DE algorithm as a Pareto-based Differential Evolution (PDE) algorithm, which is one of the new selection or optimization methods proposed in this decade. The DE is a branch of evolutionary algorithm developed by (Abbass, 2001; Price, 1995). Abbass, (2002) compared the performances of the proposed PDE optimization algorithm again 13 other existing algorithms (Abbass, 2001; Abbass, 2002). FFGA, HLGA, NPGA, NSGA, RAND, SOEA, SPEA, VEGA, PAES, PAESgray, PAES98, PAES98gray,

PAES98mut3p are the algorithms used for the statistical comparison (Abbass, 2001; Abbass, 2002). The experimentation results showed the PDE-EMO algorithm outperformed all of the other algorithms in theoretical synthesis (Abbass, 2001; Abbass, 2002). Further, Teo (2005) applied this algorithm into the legged abstract robot in order to test the performances of the algorithm used in our real life environment (Teo, 2005; Teo, 2004a; Teo, 2004b). His research proved that, the PDE-EMO can be used in generating the robot controllers that needed.

As usual, an elitist algorithm is integrated in the PDE-EMO (Abbass, 2001; Abbass, 2002). The elitism is used to generate a Pareto optimal set of ANNs that acts as a robot controller and also used to avoid losing good solutions during the EMO optimization process (Abbass, 2004; Coello, 2005a; Coello, 2005b, Zitzler, 2004; Zitzler, 2005). In the study of (Coello, 2005), they showed that elitism helps in achieving better convergence in MOEAs. Interestingly however, another argument has found with contradiction to the (Coello, 2005) research. Kukkonen (2007) experimentation results showed the selection based on ranking dominance is able to advance the search towards the Pareto-front better than the selection based on Pareto-domination. In other words, the utilization of non-elitist algorithm might produce better results compared to the elitism. This happened because the elitist algorithm always keeps the good solutions during the optimization processes, this might cause premature convergence due to high selection pressure and loss of genetic diversity. However, On (2008) study has shown the elitism outperformed the non-elitism in evolving the robots' controllers for the RF-localization behavior. The controllers generated by elitism PDE-EMO perfectly learned all of the required behaviors (On, 2008). However, Coello (2005) pointed the premature convergence problem obtained in the elitism PDE-EMO could be overcome with the utilization of archive component into the EMOs.

In the following reviewed studies (Coello, 2005; Karaboga, 2004 ; Zitzler, 2005), the authors showed an archive component could be included in the elitism in order to improve the performances of the optimization. Normally, the archive is integrated into EA in the selection process. The archive comprises only the current approximation of the Pareto set if an archive is able to be maintained during the evolution processes. The dominated archive members are removed. Thus, the best solutions are always able to be maintained during optimization processes. Furthermore, the archive used is able to overcome the premature convergence of the elitism without archive used. Interestingly however, this study has shows a contradictory experimentation results. The experimentation results showed the elitism without archive used outperformed the elitism with archive. In addition, there also have not been any studies conducted yet in comparing the elitism without archive and elitism with archive in EMO as applied to ER, hence the motivation for this investigation.

ARTIFICIAL NEURAL NETWORK

The term neural network is referred to a network or circuit of biological neurons (Haykin, 1996). Modern usage of neural network is referred to Artificial Neural Networks (ANNs). The ANNs are made up of interconnection of artificial neurons from a set of constructed executed file through programming language that mimic the properties used in biological neurons (Haykin, 1996). Neural networks are widely used for classification, approximation, prediction, and control problems (Billard, 2001; Cliff, 1993; Floreano, 1996; Floreano, 2000a; Floreano, 2000b; Horchler, 2004; Marco, 1996; Nehmzow, 1992; Nelson, 2006; Pratihar, 2003; Reeve, 2005; Teo, 2005; Urzelai, 2000). Normally, ANNs used to solve artificial intelligence problem without necessary creating a set of real biological model (Cliff,

1993; Haykin, 1996; Floreano, 2000b). Practically, the ANNs are defined as non-linear statistical data modeling and decision making paraphernalia (Haykin, 1996). Based on biological analogies, neural networks try to emulate the human brain's ability to learn from examples, learn from incomplete data and especially to generalize concepts (Haykin, 1996).

General NNs are composed of a set of nodes connected by weights. The nodes are partitioned into three layers namely input layer, hidden layer and output layer (Haykin, 1996; Teo, 2005). The neural network normally learns the behavior that required after optimization processes. A numerous of NN types are available such as Feedforward NN, Radial basis function network, Kohonen SOM network, recurrent network, stochastic NN, et al (Haykin, 1996). The feed forward NN was the first and simplest type of ANN devised (Haykin, 1996). In this network, the information moves in one direction, forward from the input nodes through the hidden nodes and to the output nodes. There are no cycles or loops involved in the network. In this study, the feedforward NN is utilized during the learning processes. The utilization of Feedforward NN is more advantageous compared to other NN due to the special features such as easy to construct, less computation time taken, and accurate used during learning processes and more importantly, feedforward NN is the most common used NN for all of the researches especially for the utilization in preliminary experiment (Billard, 2001; Cliff, 1993; Floreano, 1996; Floreano, 2000a; Floreano, 2000b; Horchler, 2004; Marco, 1996; Nehmzow, 1992; Nelson, 2006; Pratihar, 2003; Reeve, 2005; Teo, 2005; Urzelai, 2000).

A virtual Khepera robot is utilized in this study. The robot's (K-Team, 2003) light sensors, wheels, and distance sensors are enabled during phototaxis behavior evolution. Thus, there are eight distance sensors, eight light sensors and two wheels are functioning during phototaxis simulation. The distance sensors and light sensors are presented as input neurons to the ANN while the speed of the robot's wheels represents the output neurons from the ANN. Thus, there are a total number of 16 input neurons and 2 output neurons involved in the ANN during the phototaxis simulation. The Khepera robot is integrated with 8 infrared distance sensors, 1 RF receiver and 2 wheels in the RF-localization behavior simulation. The infrared distance sensors and RF receiver are presented as input neurons to the ANN while the speed of the robot's wheels represents the output neurons from the ANN. A feed-forward neural network is used as the neural controller for the robot. The chromosomes used in these experiments are a class that consists of matrix of real numbers that represent the weights used in the ANN controllers and a binary vector for the hidden layer, which represents a switch to turn a hidden unit on or off. A sigmoidal activation function is utilized for the ANN. In this work, a single hidden layer is always used and only the number of hidden nodes in this single layer is evolved. In the best outcome for the second function, one hidden neuron is still used in the controller, hence the relationship between the input and output layers are still non-linear as governed by the transformations by the sigmoidal activation function in the firing of the hidden layer as well as the output layer.

ALGORITHM USED

In this research, we attempt to solve two conflicting objectives through a single run with EMO assistance: (1) maximize the RF signal source homing behavior whilst (2) minimize the number of hidden units used in the neural controller. The RF signal source homing behavior is required in assisting the robot to home in towards the signal source. The minimization of the number of hidden nodes is done mainly to reduce the complexity of the robot's controller. This will allow for the evolved robot's behavior to

have better generalization in previously unseen environments and localization conditions. The Pareto-front thus represents a set of networks with different numbers of hidden units and different numbers of homing behaviors.

In general, DE differs from a conventional GA in a number of ways. The number of vectors used for the crossover operation is the main difference. In DE, three vectors are utilized for the crossover, where one of the selected non-dominated parents serves as the base vector to which the differences between the other two parents' vectors are added to form the trial vector. Then, the new trial vector will replace the randomly selected base vector only if it is better (Storn, 1995). The vector selection skill used in the algorithm for high behavior's fitness score with less number of hidden neurons used during the optimization processes is able to maximize the robot behavior whilst minimize the neural network complexity in the evolution process. The PDE algorithm used in this study in evolving the robot controller is presented next.

- 1.0 Begin.
- 2.0 Generate random initial population of potential chromosomes. The elements of the weight matrix are assigned random values according to a Gaussian distribution $N(0, 1)$. The elements of the binary vector ρ are assigned the value 1 with probability 0.5 based on a random generated number according to a uniform distribution between $[0, 1]$; 0 value otherwise.
- 3.0 Loop
 - 3.1 Evaluate the individuals or solutions in the population and label as parents those that are non-dominated according to the two objectives: maximizing RF-localization behavior and minimizing the number of hidden neurons.
 - 3.2 If the number of non-dominated individuals (a solution is considered as non-dominated if it is optimal in at least one objective) is less than three, repeat the 3.2.1 and 3.2.2 steps until the number of non-dominated individuals is greater than or equal to three (since the Differential Evolution algorithm requires at least three parents to generate an offspring via crossover). If insufficient solutions are retained from the first layer, then 3.2.1 and 3.2.2 steps have to be repeated for the second and subsequent layers of the non-dominated solutions.
 - 3.2.1 Find a non-dominated solution among those who are not labeled in the second layer of the non-dominated results.
 - 3.2.2 Label the solution(s) found as the non-dominated points.
 - 3.2.3 Compare among the non-dominated points with the archive solutions if it is not the first run. Otherwise, store the archive.
 - 3.2.4 Replace the non-dominated solution if archive produced higher fitness. Otherwise, store the new archive.
 - 3.3 Delete only dominated solutions from the population and retains the non-dominated solutions (elitist concept).
 - 3.4 Loop
 - 3.4.1 Select at random an individual as the main parent α_1 , and other two parents α_2, α_3 as supporting parents.
 - 3.4.2 Crossover with some uniform (0,1) probability, do

$$\omega_{ih}^{child} \leftarrow \omega_{ih}^{\alpha_1} + N(0,1)(\omega_{ih}^{\alpha_2} - \omega_{ih}^{\alpha_3})$$

if $(\rho_h^{\alpha_1} + N(0,1)(\rho_h^{\alpha_2} - \rho_h^{\alpha_3})) \geq 0.5$;
 $\rho_h^{child} \leftarrow 1$; $\rho_h^{child} \leftarrow 0$ Otherwise;

Otherwise;

$$\omega_{ih}^{child} \leftarrow \omega_{ih}^{\alpha_1} \quad \rho_h^{child} \leftarrow \rho_h^{\alpha_1}$$

And with some uniform (0,1) probability, do

$$\omega_{ho}^{child} \leftarrow \omega_{ho}^{\alpha_1} + N(0,1)(\omega_{ho}^{\alpha_2} - \omega_{ho}^{\alpha_3})$$

Otherwise;

$$\omega_{ho}^{child} \leftarrow \omega_{ho}^{\alpha_1}$$

3.4.3 Mutate with some uniform (0,1) probability, do

$$\omega_{ih}^{child} \leftarrow \omega_{ih}^{child} + N(0, mutation_rate) \quad \omega_{ho}^{child} \leftarrow \omega_{ho}^{child} + N(0, mutation_rate)$$

if $\rho_h^{child} = 0$ $\rho_h^{child} \leftarrow 1$; 0 otherwise

3.5 Repeat for all of the deleted solutions.

4.0 Repeat until maximum number of generations is reached.

5.0 End

The elitism with archive PDE-EMO algorithm listed above can be simply transform to elitism without archive PDE-EMO algorithm with removes the algorithms in 3.2.3 and 3.2.4. In equation 3.4.2, a value of 0.5 is used to give the evolving chromosome a 50% chance of either being turned on or off during the binary differential evolution crossover operation.

EXPERIMENTAL SETUP

A physics-based simulator namely WEBOTS is used in the experiments for simulating the Khepera robot's phototaxis and RF-localization behaviors. The standardized virtual Khepera robot model is ready for used in the simulator. The WEBOTS simulator has an advantage in that random noise as occurring in real Khepera robots is included for all of the light sensors, distance sensors and robot wheels (Cyberbotics, 2003a; Cyberbotics, 2003b). Thus, the robot may have slightly different responses even the same weights and inputs from the environment are used. This is highly advantageous since the simulation results represent the real-life functioning of a physical robot and more importantly, the evolved controllers are directly transferable to a real physical robot (Michel, 1998; Michel, 2004).

WEBOTS contains a rapid prototyping tool allowing the user to create 3D virtual worlds with physics-based properties, such as mass repartition, joints, friction coefficients, etc (Michel, 1998; Michel, 2004). The user can add simple inert objects or active objects called mobile robots to build the custom robots that required for any research purposes. These robots may have different locomotion schemes such as wheeled robots, legged robots or flying robots (Michel, 1998; Michel, 2004). Moreover, they can be equipped with a number of sensors and actuator receivers, like distance sensors, motor wheels, cameras, servos, touch sensors, grippers, emitters, receivers, etc (Michel, 1998; Michel, 2004). WEBOTS contains a large number of robot models and controller program examples that help the users get started such as Khepera robot model, e-puck robot model, Aibo robot model, etc (Michel, 1998; Michel, 2004).

The WEBOTS simulator included a number of robot libraries that enable user to transfer the controller programs to many commercially available real mobile robots (Michel, 1998; Michel, 2004). WEBOTS simulator also provided with some special features in simulation mode such as fast simulation, random noise simulation and supervisor capability (Michel, 1998; Michel, 2004). The integrated inherent noise feature acted as an advantage in that random noise as occurring in real robots (Cyberbotics, 2003a; Cyberbotics, 2003b; Michel, 1998; Michel, 2004).

A Khepera robot is utilized for both of the phototaxis and RF-localization simulations in this study. A Khepera robot is located on a ground with four walls. All of the walls are 30mm height and the area of the ground covered is 1m². The Khepera robot is located somewhere on the ground and facing the nearest wall. Both of the phototaxis and RF-localization simulations utilized similar general experimental setting.

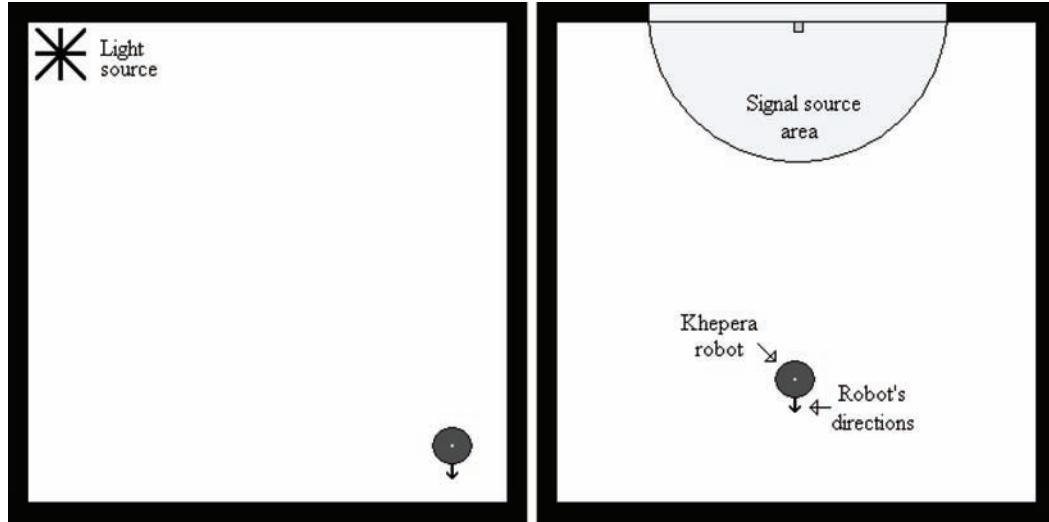
The Khepera robot's light sensors, distance sensors and wheels are enabled during phototaxis simulation whilst all of the distance sensors and wheels are activated during evolution and testing in RF-localization study. A receiver is located on the top of the Khepera robot in RF-localization simulations. It acts as a device which can receive any signal that comes from an RF emitter. An emitter is a modeled device that can send RF signals at every time step. The receiver can only detect the signal if it is within the range of emitter source; otherwise no signal is detected. The emitter source used is a radio type with buffer size 4096 and byte size 8 with radius 0.3 meter range. It is located as a static emitter near to the center of one of the walls and to the back of the Khepera robot. Whilst the light source is located at the opposite corner of the Khepera robot positioned during phototaxis behavior simulation. The general experimental setup used for both phototaxis and RF-localization simulations is depicts as in Figure 1.

The experiments conducted for the comparison among elitism with archive and elitism without archive utilized general experimental setting. Similar experimental setting used for the comparison experiment in order to ensure the experimentation outcome is not affected by the environment used. Otherwise it may cause false resulting. The comparison among the algorithms used is applied on both of the phototaxis and RF-localization behaviors simulations. This is to further ensure the comparison result is valid for different behaviors instead of stick to certain or only single behavior.

The maximum number of hidden units permitted in evolving the ANN was fixed at 15 nodes. The optimum solutions with the least possible number of hidden neurons used will be automatically generated with the utilization of the PDE-EMO algorithm. The success rate for the robot in terms of moving towards the signal source is optimal if the number of generations is set to at least 100. However, it is not necessary to evolve the robot with more than 100 generations, because the robot is already able to successfully evolve behaviors that orient towards the light and signal sources well within 100 generations. For the same reason, 60 seconds is sufficient for the simulated task duration, because the robot is also able to successfully evolve behaviors that orient towards the light and signal sources well within 60 seconds. The previous studies (On, 2008) clearly showed that the optimum solutions could be generated with 70% crossover rate and 1% mutation rate, respectively. The parameters used for the phototaxis and RF-localization behaviors evolutions are similar in order to avoid false resulting during comparison algorithms. Table 1 below depicts the parameter setting used during phototaxis and RF-localization behaviors' evolutions.

Furthermore, this study also involved the robustness study as early mentioned in the introduction section. The robustness test is divided into four levels. Different testing level indicated different number of obstacles involved. The complexity of the testing environment is increased due to the integration of number of obstacles. This happened because the most common paths used for the robot to track the

Figure 1. General experimental setup used for phototaxis and RF-localization behaviors. The left side figure represents the experimental setup used in phototaxis simulation whilst the right side figure represents the experimental setup used in RF-localization behavior simulation.



light source and signal source are indirectly blocked by the inclusion of obstacles. The robustness tests are purposely conducted for the controllers' performances and effectiveness tests. Figure 2 (a) and 2 (b) below represent different level of environments used in robustness test for phototaxis and RF-localization behaviors, respectively.

TEST FUNCTION USED

A number of preliminary tests had been carried out in order to obtain the most suitable fitness functions used for the Khepera robot phototaxis and RF-localization behaviors. As a result, a combination of several criteria into the fitness functions is proposed from the preliminary experimentation results. The phototaxis evaluation function used is shown in F_p ; (1). Fitness function, F_p encompasses of terms that demonstrate maximize the average robot wheels speed, and maximize the light following behavior. While the fitness function used for the RF-Localization; F_{RF} ; (2), comprises of terms that demonstrate obstacle avoidance behaviors, maximize the average speed of the robot's wheels, maximize the robot wheels speed and lastly maximize the robot RF-localization behavior. The additional fitness function involved in this study is shown in (3). The corresponding fitness function represents minimization for the number of hidden neurons used during the optimization processes. The formulation of the fitness functions are as follows:

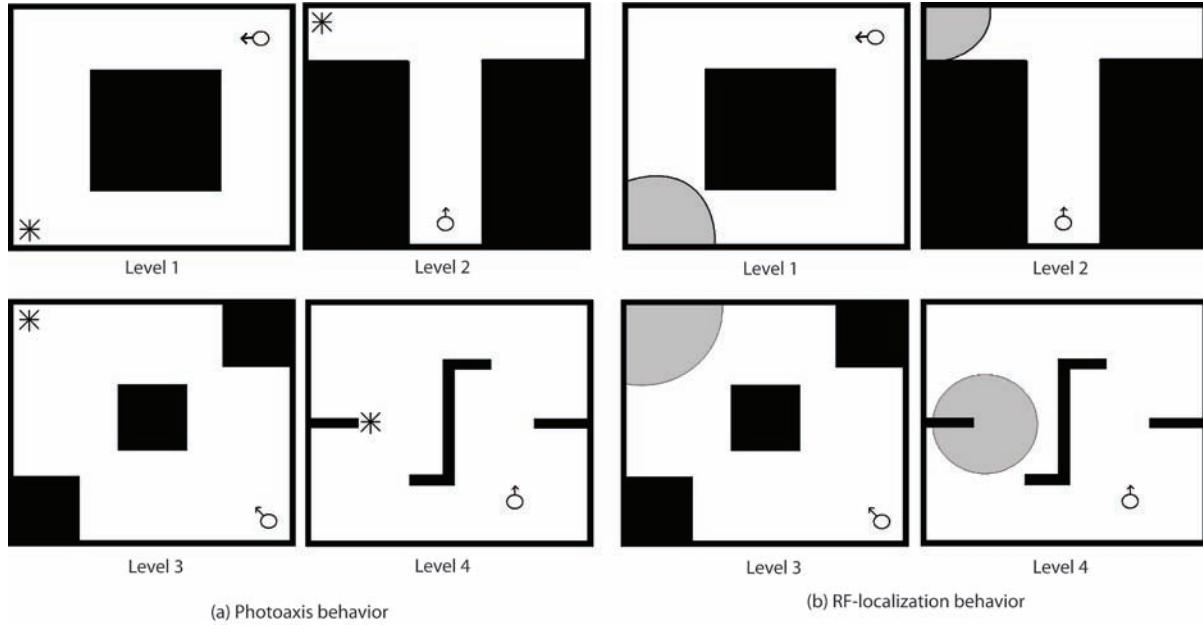
$$F_p = V * (1 - \text{sqrt}(DV)) * (1 - L) \quad (1)$$

$$F_{RF} = \frac{1}{T} \sum_{t=0}^T (1 - i) S V W_L W_R \quad (2)$$

Table 1. Parameter used for evolutions

Descriptions	Parameters
Number of Generation	100
Population Size	30
Simulation Time Steps	60s
Crossover Rate	70%
Mutation Rate	1%
Hidden Neurons	15
Repeated Simulations	10
Random Noise Feature	Activated

Figure 2, Different level of environments used in phototaxis and RF-localization behaviors' tests.



$$0 \leq (1 - i) \leq 1$$

$$S = [1, 50]$$

$$0 \leq V \leq 1$$

$$0 \leq W_L, W_R \leq 1$$

$$F_{HN} = \sum_{c=0}^C H_c \quad (3)$$

Where F represents the fitness function, T = simulation time, i = highest distance sensor activity, L = the value of the proximity sensor with the highest activity (*light following behavior*), DV represents the algebraic difference between the wheels speed, V = average speed of wheels, S = signal source value (*RF localization behavior*), W_L = left wheel speed, W_R = right wheel speed and H = hidden neuron used, with $c = 1 \dots 15$ representing the number of the corresponding hidden neuron.

The fitness values of F_{RF} and F_p are accumulated during the life of the simulated robot and then divided by the simulation time. The average wheels speed is one of the most critical components involved in the phototaxis behavior. The controller always has to first evolve the behavior to avoid slow movement before it can perfectly learn to track the light intensity. Otherwise, it is a difficult task in optimizing the phototaxis behavior due to low accumulated fitness values generated from slow robot movement. For the RF-localization behavior, the obstacle avoidance component plays as one of the most important components in the experiment since the Khepera robot is evolved with the initial orientation of facing away from the signal source. Furthermore, the robot can only detect the signal if it is within the range of emitter source; otherwise no signal is detected. Thus, the controller always has to first evolve a behavior to avoid crashing into the opposite wall that it starts facing towards before it can home towards the RF signal source. This differs to the phototaxis behavior. The obstacle avoidance component is optional during phototaxis behavior evolution. The robot could avoid from bumping to the nearest walls even the obstacles avoidance component is not included in the phototaxis evolution. The second important component in the F_{RF} function is the signal source behavior, where the Khepera robot must allocate the source properly and attempt to stay in the source area if possible. The other components are used to avoid the robot from evolving to achieve the target but without a spinning movement that uses more time to localize towards the signal source. F_{HN} represents the numbers of hidden neurons required and are used to reduce the complexity of the neural structure of the robot's controller.

The fitness function F_p that is used in this study is the standard fitness function used to evolve and evaluate potential controllers for phototaxis behavior which was originally proposed by Floreano, (2000a, 2000b) and subsequently adopted as the common method of fitness assignment for evolving light-following behavior in two-wheeled autonomous mobile robots. However, the fitness function F_p was modified and adapted into F_{RF} to evolve for the RF-localization behavior. A number of preliminary tests had been carried out in order to obtain the most suitable fitness functions used for the Khepera robot for the successful evolution of RF-localization behavior (On, 2008).

ALGORITHMS PERFORMANCES COMPARISON

Phototaxis Behavior

In this experiment, the elitism with archive and elitism without archive PDE-EMO algorithms were utilized to generate a Pareto optimal set of ANN that optimizes two conflicting objectives for robot's phototaxis behavior. The conflicting objectives involved maximizing the Khepera robot's behavior for tracking the light source and minimizing the neural network complexity by reducing the hidden neurons required in its controller. Typically, elitism is used to avoid losing good solutions during EMO optimization process. However, there is no study conducted yet in comparing again the performances between the application of elitism with archive and elitism without archive in ER field. Hence, the motivation is investigation.

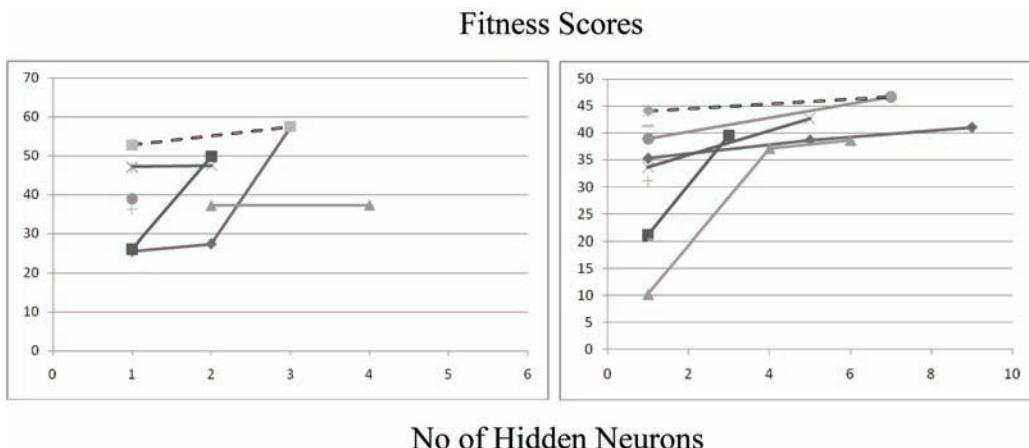
The simulation results showed there were no failed repeated simulations in evolving the robot controllers using elitism with archive and elitism without archive for phototaxis behavior. In most of the cases, only few hidden neurons are involved in the Pareto solutions. Figure 3 depicts the obtained local Pareto-front solutions from all of the repeated simulations and the global Pareto solutions found with respectively.

In terms of solution preservation, Figure 3 (left) has clearly shows that there are two global solutions obtained from the phototaxis behavior's evolution using elitism without archive algorithm. The first Pareto solution obtained utilized only one hidden neuron with 52.75 fitness score whilst the second Pareto solution found scored 57.52 fitness value with utilized 3 hidden neurons. Furthermore, Figure 3 (right) shows that there are also two solutions obtained from the phototaxis behavior's evolution using elitism with archive algorithm. The global Pareto solutions found involved one and seven hidden neurons respectively with 44.12 and 46.67 fitness scores. The evidence has clearly shows that the ANN complexity was able to be minimized due to the utilization of either elitism with archive or elitism without archive PDE-EMO in evolving the robot controllers since fairly successful controllers could be produced using only few hidden neurons. These controllers allowed the robots to navigate towards the light source with the lowest number of hidden neurons found in the Pareto set. Nevertheless, the results show that the utilization of elitism without archive algorithm is outperformed the utilization of elitism with archive algorithm in this experiment.

Tests were conducted for all of the generated controllers to verify their ability in tracking the light source. A comparison of the average time taken and average success rate was conducted for all of the generated robot's controllers. Each of the evolved controllers was tested with the same environmental setting as that used during evolution for 3 times. The testing results obtained during the tests were tabulated as Table 2 shows below.

Table 2 (a) has shows that most of the tests performed achieved 100 of success rate even 3 out of 10

Figure 3. Obtained simulation results. The local and global Pareto-frontier for all of the generated controllers from repeated simulations are illustrated. Dark double dotted-line represents the global Pareto solutions. (left) Pareto curve for all of the repeated phototaxis behavior simulations using elitism without archive, (right) Pareto curve for all of the repeated phototaxis behavior simulations using elitism with archive



trials performed non-perfect average success rate and Table 2 (b) shows that most of the tests performed using elitism with archive for 4 out of 10 trials have achieved 100% of success rate. Furthermore, the testing results also show that the average success rate obtained in elitism without archive is higher than the average success rate obtained in elitism with archive, 95.93% and 91.70% with respectively. The evidence has clearly shown that the controllers' evolved using elitism without archive outperformed the elitism with archive. Furthermore, the evidence of the performances' comparison also can be simply demonstrated via robots' movements. Figure 4 illustrates the robots' movements obtained via comparison testing results.

Figure 4 (b) shows that, in some tests, robot might fail to perform the task with successfully due to the navigation and collision avoidance problem. The collision avoidance component was not included in the evaluation function. Thus, the robot might fail to navigate towards the light source if the robot was stuck or bumped to the wall located nearby during tracking the light source. The robot also might fail to navigate towards the light source area if the robot bumped to the wall located nearby. Thus, it caused high failed results. Contradictory results obtained from the controllers evolved using elitism without archive. Figure 4 (a) shows that, the robots were able to navigate and track the light source successfully with straight line movement. The robots also were able to navigate with high speed and thus archived the light source faster compared to the robots evolved using elitism with archive. Furthermore, Figure 4 (a) also shows that the robots were able to stay as longer as possible in the signal source area without bumping to the wall. As a simple conclusion, we observed that the controllers evolved using elitism without archive is outperformed the controllers evolved using elitism with archive.

RF-Localization Behavior

The conflicting objectives in this experiment involved maximizing the Khepera robot's behavior for homing towards the signal source and minimizing the hidden units required in its controller. In (Kim

Table 2. Testing results obtained for the entire generated controllers using elitism with archive and elitism without archive, respectively.

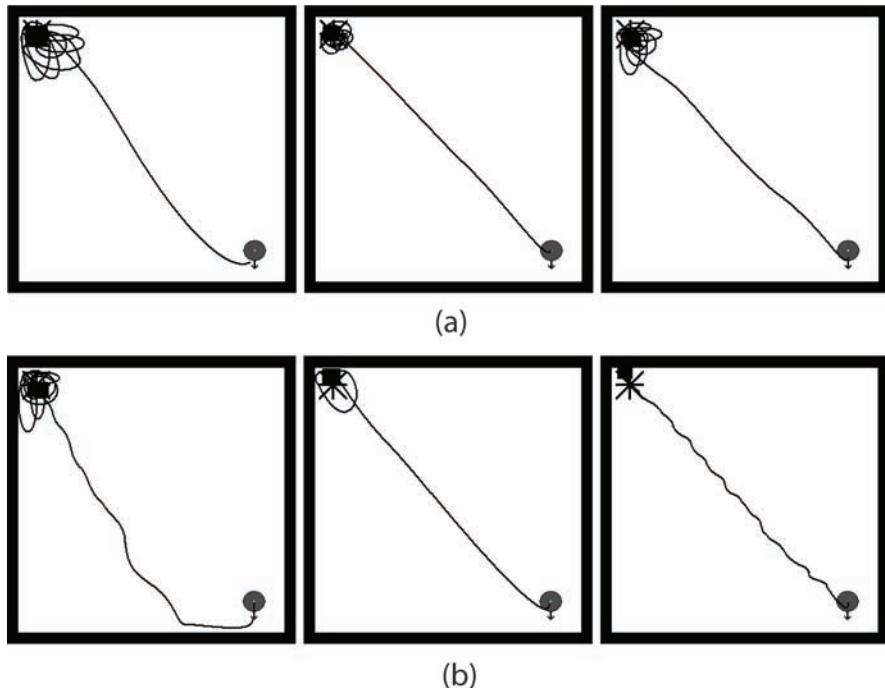
Elitism without archive (a)			Elitism with archive (b)		
Trial	Average Success Rate (%)	Average Time Taken (s)	Trial	Average Success Rate (%)	Average Time Taken (s)
1	100	22.567	1	87.65	28.667
2	100	24.567	2	100	32.567
3	100	21.546	3	100	32.236
4	74.07	28.553	4	69.69	30.165
5	97.53	29.665	5	87.65	29.667
6	100	22.161	6	100	31.256
7	100	23.168	7	96.67	31.567
8	100	23.154	8	87.65	28.667
9	87.65	27.350	9	100	30.159
10	100	23.567	10	87.65	28.667
Average	95.93	24.630	Average	91.70	30.362

On, 2008) study, it was clearly shown that elitism helps in achieving better convergence in EMOs. The elitist PDE-EMO is used to generate a Pareto optimal set of ANNs that acts as a robot controller. Nevertheless, the comparison among elitism with archive and elitism without archive is not investigated in that study.

The simulation results showed there were no failed repeated simulations in evolving the robot's RF-localization behaviors for both of the algorithms. In most of the cases, only a hidden neuron is involved in the Pareto solution. The robot could perform the task very fast and efficient. Hence, we observed that the utilization of PDE-EMO algorithm has successfully minimized the number of hidden neurons used in evolving the robot controller for the autonomous mobile robots' RF-localization behavior. Figure 5 represents the obtained local Pareto-front solutions from all of the 10 repeated simulations and the global Pareto solutions found.

Figure 5 (left) shows that the optimum solutions were able to be maintained successfully in the simulations. The global Pareto-frontier found in at the last generation also utilized very few hidden neurons. The solutions involved only 1, 2, and 5 hidden neurons out of the permissible 15 hidden neurons in the evolution. Again, this shows that the ANN complexity was able to be minimized due to the utilization of elitism PDE-EMO in evolving the robots to home in towards the signal source area even with the lowest number of hidden neurons found in the Pareto set. Furthermore, Figure 5 (right) also shows that the optimum solutions were able to be maintained successfully in the simulations. The global Pareto found in at the last generation only utilized 1 and 4 hidden neurons out of the permissible 15 hidden neurons in the evolution. Nevertheless, the results show that the controllers evolved using elitism without archive utilized less hidden neurons and obtained higher fitness scores compared to the results obtained from

Figure 4. Obtained robot movements from phototaxis behavioral tests using elitism with archive (b) and elitism without archive (a) algorithms.



the controllers evolved using elitism with archive. Thus, we observed that, the controller evolved using elitism without archive is slightly outperformed the controller evolved using elitism with archive.

In some cases, the second function might have been very effectively optimized, resulting in controllers with very small numbers of hidden neurons. It might hence consist of only a single hidden neuron that is used by the evolved controller, such as the global Pareto solutions obtained as shown in Figure 5 (left). The robot is still able to perform the task and accomplish the RF-localization objective (albeit with a lower task behavior optimality) with just a single hidden neuron.

Tests were conducted for all of the generated controllers to verify their ability in tracking the signal source. A comparison of the average time taken and average success rate was conducted for all of the generated robot's controllers. Each of the evolved controllers was tested with the same environmental setting as that used during evolution for 3 times. The testing results obtained during the tests are tabulated as Table 3 shows above.

Table 3 (a) shows most of the robots' controllers generated using elitism without archive achieved 100 of successful rate even 4 out of 10 trials performed non-perfect average success rate. However, Table 3 (b) shows none of the controllers generated by elitism with archive achieved 100 of success rate. The testing results show that the robots' controllers generated using elitism without archive achieved 85.73 average success rates. Whilst the robots' controllers generated using elitism with archive achieved 66.16 average success rates. Thus, we observed that the robots' controllers generated using elitism without archive are outperformed the robots' controllers generated using elitism with archive. Further, the evidence of the proof is included in Figure 6. Figure 6 shows the robots' movements generated by all of the evolved controllers.

Figure 5. Obtained simulation results. The local and global Pareto-frontier for all of the generated controllers from repeated simulations are illustrated. Dark double dotted-line represents the global Pareto solutions. (left) Pareto curve for all of the repeated RF-localization behavior simulations using elitism without archive; (right) Pareto curve for all of the repeated RF-localization behavior simulations using elitism with archive.

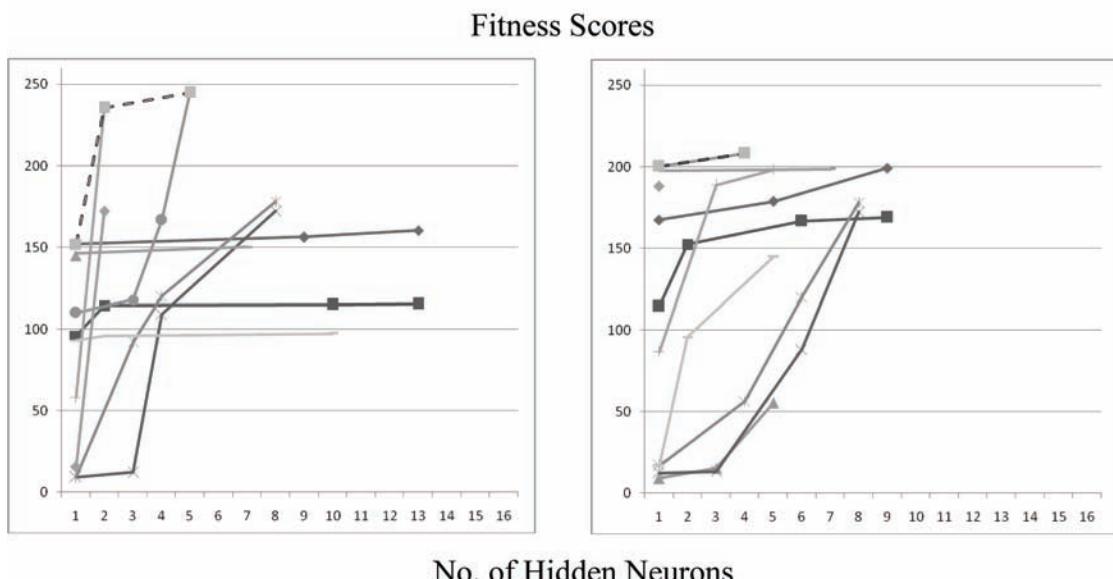


Figure 6 (a) shows that, the robots were able to navigate and track the signal source successfully with straight line movement. The robots also were able to navigate with high speed and thus achieved the signal source faster compared to the robots evolved using elitism with archive. Figure 6 (a) also shows that the robots were able to stay as longer as possible in the signal source area without bumping to the wall. Figure 6 (b) shows that, in some tests, robot might fail to perform the task with successfully due to the navigation problem. The robot also might fail to navigate to the signal source area if the robot was bumped to the wall located nearby. This is probably due to the fact that some of the controllers allowed the robot to turn only to the left hand side when the robot moved near to the wall. Thus, the robot might take longer distances as well as a longer amount of time to track the signal source, hence reducing its fitness scores considerably. Consequently, it caused high failed results. As a simple conclusion for this RF-localization behavior's experiment, we observed that the controllers evolved using elitism without archive is outperformed the controllers evolved using elitism with archive.

ROBUSTNESS TESTS

Tests were conducted for all of the generated controllers in order to verify their ability in tracking the light source and signal source with robustly. The controllers evolved with elitism without archive were utilized in the tests. The comparison results showed the controllers evolved with elitism without archive outperformed the controllers evolved by elitism with archive. The testing results showed that most of the individuals were able to achieve the target with very few hidden units used. The robot achieved the objectives with different paths, different time steps, and different movements even with the same number of hidden units used. A comparison of the time taken and success rate was conducted for all of the generated robot's controllers. Each of the evolved controllers was tested with the different level of environments for 3 times. The comparison of the tested controllers is depicted as in Table 4.

Table 3. Testing results obtained for the entire generated controllers using elitism with archive and elitism without archive, respectively in RF-localization behavior.

Elitism without archive (a)			Elitism with archive (b)		
Trial	Average Success Rate (%)	Average Time Taken (s)	Trial	Average Success Rate (%)	Average Time Taken (s)
1	86.67	18.669	1	66.67	33.496
2	100	19.267	2	86.67	30.259
3	100	9.873	3	33.33	33.679
4	43.33	38.667	4	66.67	34.367
5	37.67	41.567	5	44.59	33.679
6	100	9.973	6	66.67	31.037
7	100	20.006	7	77.00	31.379
8	100	13.567	8	66.67	33.496
9	89.67	18.667	9	66.67	30.895
10	100	16.421	10	86.67	30.158
Average	85.73	20.668	Average	66.16	32.245

Figure 6. Obtained robot movements from RF-localization behavioral tests using elitism with archive (b) and elitism without archive (a) algorithms.

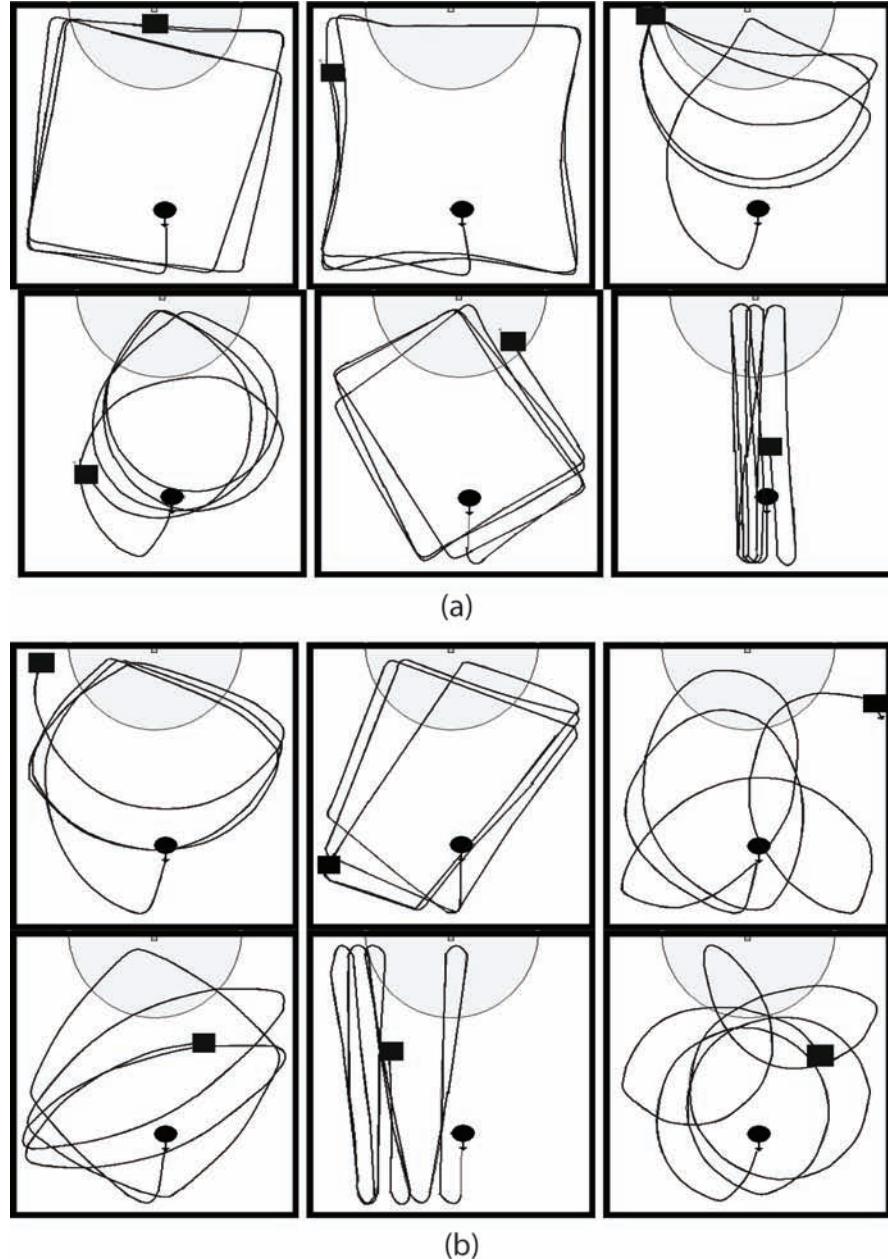


Table 4 shows that, the average success rate achieved for both of the behaviors' test were slightly reduced due to the increment of obstacles used during testing phases. Similar case happened for the average time taken for the robots to achieve their targets.

The comparison results show that the robots were not as robust to the environment when tests were performed in the Maze level 4 compared to other mazes. This is probably due to the fact that some of the controllers allowed the robot to turn only to the left hand side when the robot moved near to the

Table 4. Comparison of average success rate and time taken for all of the generated controllers from the 10 repeated simulations for phototaxis and RF-localization behaviors.

Light Following Behavior		
Testing Environment	Average Success Rate (%)	Average Time Taken (%)
Maze Level 1	70.67	30.671
Maze Level 2	68.67	33.287
Maze Level 3	62.33	38.667
Maze Level 4	33.33	42.432
RF-Localization Behavior		
Testing Environment	Average Success Rate (%)	Average Time Taken (%)
Maze Level 1	45.67	28.781
Maze Level 2	40.33	30.544
Maze Level 3	18.67	55.670
Maze Level 4	9.79	58.730

wall. Thus, the robot might fail to track the light and signal source with correctly. In addition, the Maze level 4 consists of five obstacles that blocked the most common paths used for the robot to track the light and signal sources. The maze used in Maze level 4 is the most complex among all of the mazes involved in the robustness tests.

Furthermore, the average success rate for the phototaxis tests were still higher compared to the RF-localization behavior tests. This happened due to the fact that the Khepera robots were able to track the light source easily compared to explore for the signal source. The Khepera robot's light sensors were able to capture the light intensity value even if the Khepera robot was positioned hiding in a corner. However, the Khepera robot's receiver may not be able to sense any value if the robot failed to explore and track for the signal source. Thus, the average success rate obtained for the light following behavior is logically to be higher compared to the RF-localization average success rate.

In some of the outstanding cases, the evolved robots flawlessly learned to move with a straight line path either to track the light source or home in towards the signal source. Interestingly, one of the evolved controllers surprisingly learned wall-following to allocate the signal even though this was never explicitly evaluated for in the fitness function used. But, the robots took slightly more time to home in on the signal source because the controllers did not allow the robot to directly rotate backwards to move towards the signal. Nevertheless, this was not happened for the light following behavior test. None of the evolved phototaxis robot controller learned to navigate and track the light source with wall following behavior. Figures 7 and 8 depicts some of the observed robot movements from the testing phase, and show a comparison of robot movements for the light following (Figure 7) and RF-localization (Figure 8) behaviors in different testing environments.

Figure 7 has shown that, the robots could track the light source more habitually compared to the robots evolved for RF-localization behavior. In other words, the robots were able to track for the light source easily compared to home in towards the signal source. Robot's exploration behavior acted as an important component in the RF-localization behavior. The robots took slightly more time to home in on the signal source because the controllers must first successfully track for the signal source. In addition, the controllers did not allow the robot to directly rotate backwards to move towards the signal once

the receiver sensed emitter source. In Figure 7 (b), it was observed that most of the robots appeared to move randomly in exploring the environment to find the signal source. Some of the controllers allowed the robot to turn only to the left hand side when the robot moved near to the wall. Thus, the robot might spin towards in tracking the signal source. It managed to approach the wall where the signal source was located once it found the signal source. Contradictory results obtained from Figure 7 (a). It was observed that most of the robots appeared without randomly track for the light source. The robots were able to track for light source even they were position far away from the source. However, some of the failed controllers showed the obstacles avoidance behavior was more important in the mazes tests. The robot might fail to navigate to the light source if it was failed to avoid from bumping to the nearest wall or obstacles. The robot managed to navigate towards the light source without bumping into the obstacles nearby in some cases.

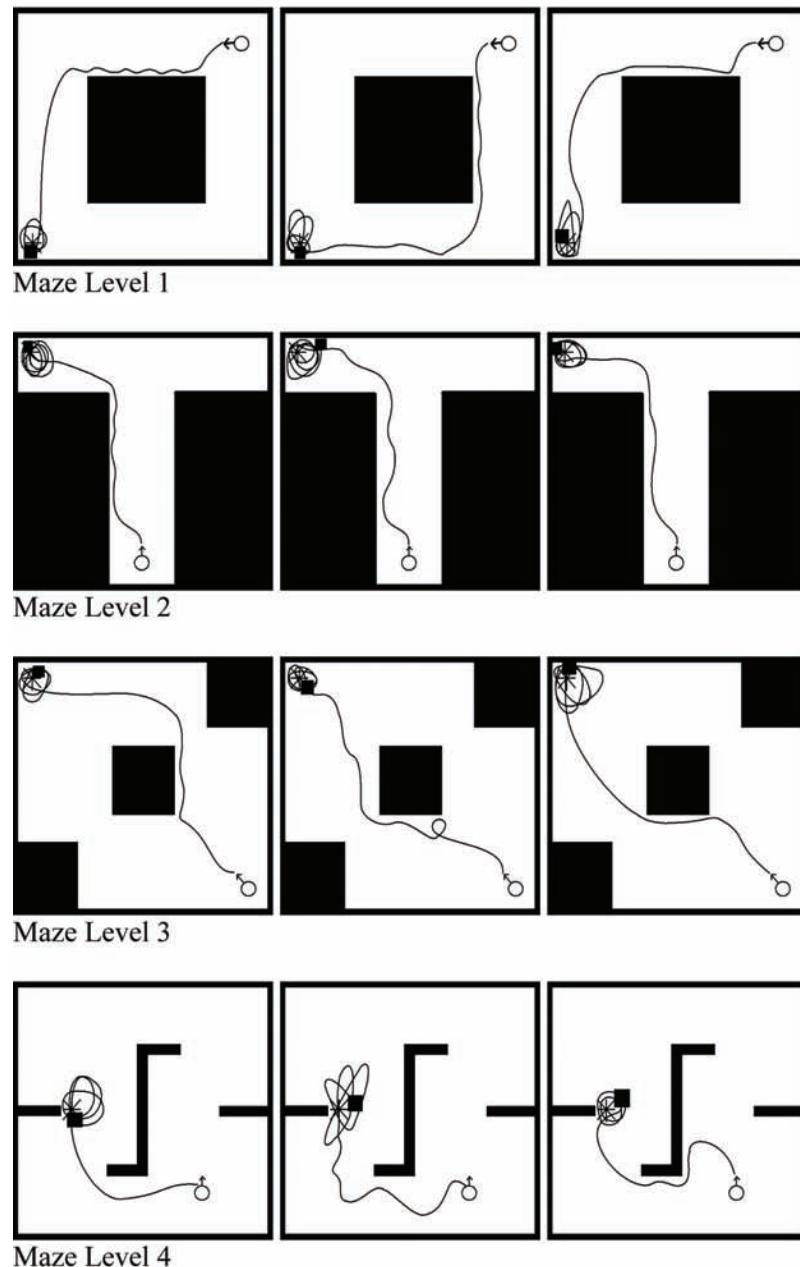
CONCLUSION & FUTURE WORK

From the evolutionary and empirical experiments conducted, the robot controllers could be successfully generated by the PDE-EMO algorithm. The comparison between elitism without archive and elitism with archive was conducted. Both of the methods used were able to generate the required controllers, even though the evolutionary optimization processes were conducted in the presence of noise. However, the utilization of elitism without archive performed slightly better than elitism with archive. The evidence was depicted in the comparison experimentation results. The most suitable fitness functions used in evolving the robot controllers for phototaxis and RF-localization were also successfully identified. The robustness was tested for the best performing controllers at four different levels. It was observed that the controllers were robust to the environment even when the robots were tested and simulated in environments different to the original evolution environment. The obstacles' positioning only introduced minor effects to the controllers' ability to carry out their required behaviors. Nevertheless, there are still elements that can be further improved in the PDE-EMO approach for ER.

More obstacles could be included in future experimental environments in order to evolve controllers that can be used in more complex environments. The incremental evolution concept may also be considered to be used in the evolution process to start the evolution of robot controllers to perform additional new tasks with best individuals obtained from the previous task evolution. The fitness function used plays an important role in evolving the robot controller, which is not something trivial to design. Hence, a co-evolutionary approach might also be beneficial in more complex environments where a suitably successful fitness function may be hard to identify manually.

As a conclusion for this experimental study, it has been shown that the PDE-EMO could successfully produce controllers with phototaxis and RF-localization behaviors while minimizing the number of hidden neurons used in evolving the controllers. We observed that even controllers with very few hidden neurons used could perform the task successfully and also that the evolved solutions were generally robust to the environments that are distinctly different during the evolution and testing phases. Furthermore, the utilization of elitism without archive outperformed the utilization of elitism with archive in evolving the robot controllers.

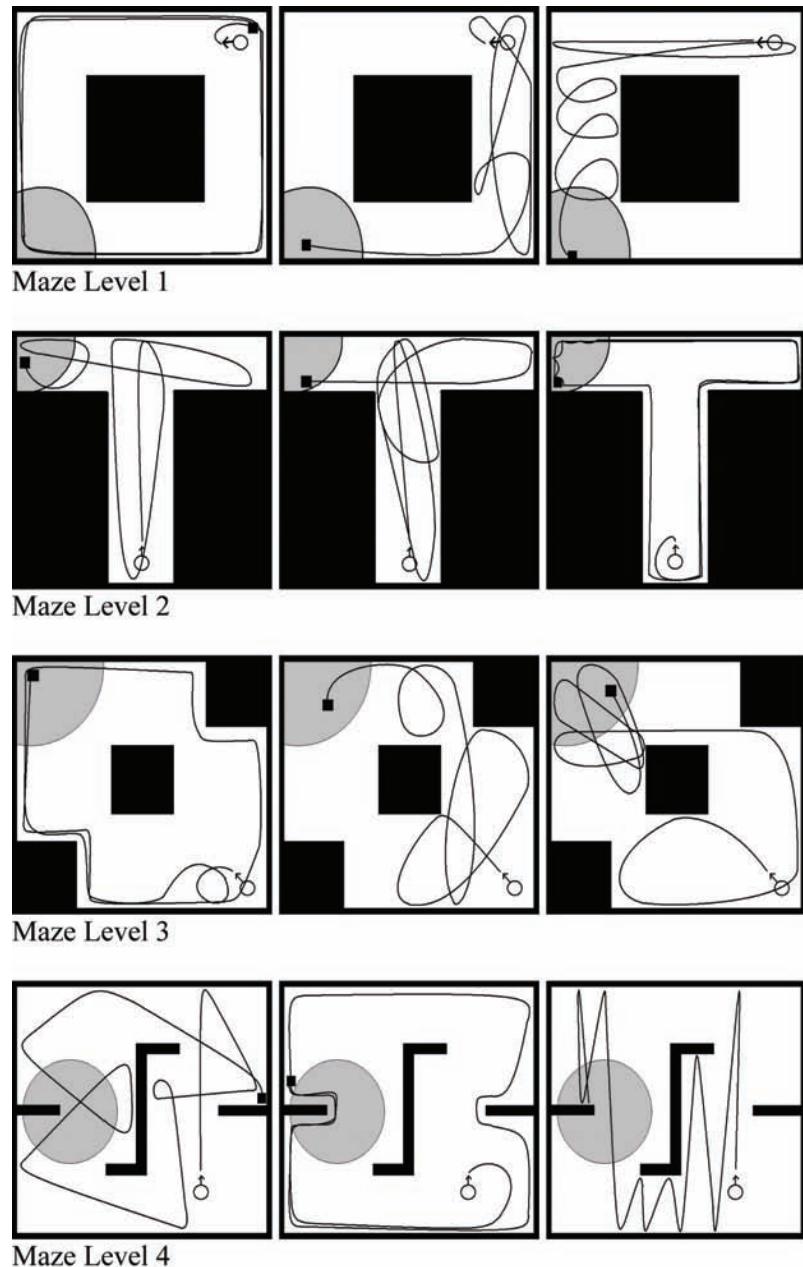
Figure 7. Phototaxis robustness tests.



REFERENCES

- Abbass, H. A., Ruhul, S., & Newton, C. (2001). PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems. In *Proceedings of the Congress on Evolutionary Computation (CEC'2001)*, IEEE Service Center, Piscataway, NJ (pp. 971-978).

Figure 8. RF-localization robustness test.



Abbass, H. A., & Sarker, R. (2002). The Pareto differential evolution algorithm. *International Journal of Artificial Intelligence Tools*, 11(4), 531–552. doi:10.1142/S0218213002001039

Alba, E., & Tomassini, M. (2002). Parallelism and evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 6(5), 443–462. doi:10.1109/TEVC.2002.800880

- Billard, A., & Schaal, S. (2001). A connectionist model for online robot learning by imitation. In *Proceedings of the 2001 IEEE International Conference on Intelligent Robots and Systems*.
- Cliff, D., Harvey, I., & Husbands, P. (1993). Explorations in evolutionary robotics. *Adaptive Behavior*, 2(1), 73–110. doi:10.1177/105971239300200104
- Coello, C. A. C. (2005a). Recent trends in evolutionary multiobjective optimization: Theoretical advances and applications. In *Evolutionary multiobjective optimization* (pp. 7-32). Berlin, Germany: Springer-Verlag.
- Coello, C. A. C., Pulido, G. T., & Montes, E. M. (2005b). Current and future research trends in evolutionary multiobjective optimization. In *Information processing with evolutionary algorithms* (pp. 213-231). Berlin, Germany: Springer.
- Cyberbotics. (2003a). *Distance sensors random noise*. Retrieved from <http://www.cyberbotics.com/cdrom/common/doc/webots/reference/section2.15.html>
- Cyberbotics. (2003b). *WEBOTS simulator*. Retrieved from <http://www.cyberbotics.com/>
- Deb, K. (2007). Evolutionary multi-objective optimization without additional parameters. *Parameter Setting in Evolutionary Algorithm. Studies in Computational Intelligence*, 54, 241–257. doi:10.1007/978-3-540-69432-8_12
- Floreano, D., & Mondada, F. (1996). Evolution of homing navigation in a real mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26, 396–407. doi:10.1109/3477.499791
- Floreano, D., & Urzelai, J. (2000a). Evolutionary on-line self-organization of autonomous robots. In M. Sugisaka (Ed.), *Proceedings of the 5th International Symposium on Artificial Life and Robotics (AROB'2000)*, Oita, Japan.
- Floreano, D., & Urzelai, J. (2000b). Evolutionary robotics: The next generation. In T. Gomi (Ed.), *Proceedings of Evolutionary Robotics III: From Intelligent Robots to Artificial Life*. Ontario, Canada: AAI Books.
- Gibson, R. (2007). *Radio and television reporting*. Retrieved from http://www.fdv.uni-lj.si/Predmeti/Dodiplomski/Arhiv/ucni_nacrti_2002-03.pdf
- Haykin, S. (1999). *Neural networks: A comprehensive foundation*. Upper Saddle River, NJ: Prentice Hall.
- Horchler, A., Reeve, R., Webb, B., & Quinn, R. (2004). Robot phonotaxis in the wild: A biologically inspired approach to outdoor sound localization. *Journal in Advanced Robotics*, 18(8), 801–816. doi:10.1163/1568553041738095
- K-Team. (2003). *Khepera robot*. Retrieved from <http://www.k-team.com>
- Karaboğa, D., & Ökdem, S. (2004). A simple and global optimization algorithm for engineering problems: differential evolution algorithm. *Turk J Elec Engin*, 12(1).
- Kukkonen, S. (2007). Ranking-dominance and many-objective optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2007)*, Singapore (pp. 3983-3990).

- Marco, C., Dorigo, M., & Borghi, G. (1996). Behavior analysis and training – a methodology for behavior engineering. *IEEE Transactions on Systems, Man, and Cybernetics*, 26(6), 365–380.
- Michel, O. (1998). Webots: Symbiosis between virtual and real mobile robots. In *Proceedings of the First International Conference on Virtual Worlds (VW'98)*, Paris, France (pp. 254-253). Berlin, Germany: Springer Verlag.
- Michel, O. Cyberbotics, Ltd. (2004). WebotsTM: Professional mobile robot simulation. *International Journal of Advanced Robotic Systems*, 1(1), 39–42.
- Nehmzow, U. (1992). *Experiments in competence acquisition for autonomous mobile robots*. Unpublished doctoral dissertation, Dept of Artificial Intelligence, University of Edinburgh.
- Nelson, A. L., & Grant, E. (2006). Developmental analysis in evolutionary robotics. In *Proceedings of the Adaptive and Learning System, IEEE Mountain Workshop on IEEE CNF* (pp. 201-206).
- NOAA Satellite & Information Service. (2002). *Search, rescue and tracking*. Retrieved from <http://www.sarsat.noaa.gov/>
- Nolfi, S., & Floreano, D. (2000). *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. Cambridge, MA: The MIT Press.
- On, C. K., Teo, J., & Saudi, A. (2008). Multi-objective artificial evolution of RF-localization behavior and neural structures in mobile robots. In *Proceedings of the IEEE World Congress on Computational Intelligence (WCCI-CEC 2008)*, Hong Kong.
- Pike, J. (2003). *GPS overview from the NAVSTAR joint program office*. Retrieved from <http://www.fas.org/spp/military/program/nav/gps.htm>
- Pratihar, D. K. (2003). Evolutionary robotics – a review. *Sadhan.*, 28(6), 999–1009. doi:10.1007/BF02703810
- Reeve, R., Webb, B., Horchler, A., Indiveri, G., & Quinn, R. (2005). New technologies for testing a model of cricket phonotaxis on an outdoor robot platform. *Robotics and Autonomous Systems*, 51(1), 41–54. doi:10.1016/j.robot.2004.08.010
- Storn, R., & Price, K. (1995). *Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces* (Tech. Rep. No. TR-95-012). International Computer Science Institute, Berkeley.
- Teo, J. (2005). Evolutionary multi-objective optimization for automatic synthesis of artificial neural network robot controllers. *Malaysian Journal of Computer Science*, 18(2), 54–62.
- Teo, J., & Abbass, H. A. (2004a). Embodied legged organisms: A Pareto evolutionary multi-objective approach. *Journal of Evolutionary Computation*, 12(3), 355–394. doi:10.1162/1063656041774974
- Teo, J., & Abbass, H. A. (2004b). Multi-objectivity and complexity in embodied cognition. *IEEE Transactions on Evolutionary Computation*, 9(4), 337–360. doi:10.1109/TEVC.2005.846902

- Urzelai, J., & Floreano, D. (2000). Evolutionary robots with fast adaptive behavior in new environments. In *Proceedings of the Third International Conference on Evolvable Systems: From Biology to Hardware (ICES2000)*, Edinburgh, Scotland.
- Zitzler, E., Bleuler, S., & Laumanns, M. (2004). A tutorial on evolutionary multiobjective optimization. In X. Gandibleux, et al. (Eds.), *Metaheuristics for multiobjective optimization*. Berlin, Germany: Springer.
- Zitzler, E., Deb, K., & Thiele, L. (2005). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8.

KEY TERMS AND DEFINITIONS

Artificial Neural Networks: is a computational model based on biological neural networks which consists of an interconnected group of artificial neurons that have been practically used in prediction, classification, control problem and approximation. In other words, ANNs are an adaptive system that changes its structure based on internal or external information that flows through the network during the learning phase. ANNs consist of a set of nodes (input, hidden and output neurons) connected by weights.

Autonomous Mobile Robots: are robots that have the capability to navigate using their integrated sensors and wheel(s) independently using its onboard controllers as well as able to perform the requested tasks in an unstructured environment.

Evolutionary Computation: refers to a subfield of artificial intelligence or computational intelligence that involves computational algorithms that are inspired by biological processes.

Evolutionary Multi-Objective Optimization: is known as multi-attribute or multi-criteria evolutionary optimization that involves simultaneous processes that optimize two or more conflicting objectives which may be subject to certain constraints.

Evolutionary Robotics: is a method that utilizes evolutionary computation to automatically synthesize controllers for autonomous robots. In other words, it refers to the application of artificial evolution to generate autonomous robots and/or their controllers with minimal or no direct input from humans.

Pareto Differential Evolutionary Algorithm: is a term that refers to hybridization of Evolutionary Multi-objective Optimization (EMO) into the Differential Evolution algorithm (DE).

Phototaxis Behavior: is a kind of taxis behavior that occurs when a whole organism navigates or tracks in response to a light stimulus. The movement of the organism in the direction of light is defined as positive. Otherwise, it is negative. This is advantageous for phototrophic organisms to orient themselves towards light sources to receive energy for photosynthesis. This behavior is applied to the robotics area as a homing behavior.

Radio Frequency Localization Behavior: is a kind of taxis behavior that occurs when an entity navigates in response to the propagation of radio frequency. This behavior is applied to the robotics area as an ideal Search and Rescue (SAR) approach.

Chapter 29

Improving Automated Planning with Machine Learning

Susana Fernández Arregui
Universidad Carlos III de Madrid, Spain

Sergio Jiménez Celorio
Universidad Carlos III de Madrid, Spain

Tomás de la Rosa Turbides
Universidad Carlos III de Madrid, Spain

ABSTRACT

This chapter reports the last machine learning techniques for the assistance of automated planning. Recent discoveries in automated planning have opened the scope of planners, from toy problems to real-world applications, making new challenges come into focus. The planning community believes that machine learning can assist to address these new challenges. The chapter collects the last machine learning techniques for assisting automated planners classified in: techniques for the improvement of the planning search processes and techniques for the automatic definition of planning action models. For each technique, the chapter provides an in-depth analysis of their domain, advantages and disadvantages. Finally, the chapter draws the outline of the new promising avenues for research in learning for planning systems.

INTRODUCTION

Automated Planning (AP) is the branch of Artificial Intelligence (AI) that studies the computational synthesis of sets of actions to carry out a given task (Ghallab et al., 2004). AP appeared in the late '50s from converging studies into state-space search, theorem proving and control theory to solve the practical needs of robotics. The STanford Institute Problem Solver STRIPS (Fikes & Nilsson, 1971), developed to be the planning component for controlling the autonomous robot Shakey (Nilsson, 1984), perfectly illustrates the interaction of these influences. Since the Shakey's days up to now, AP has created efficient planners and well accepted standards for representing and solving the planning tasks. In the last

DOI: 10.4018/978-1-60566-766-9.ch029

years, AP systems have been successfully applied for controlling underwater vehicles (McGann et al., 2008), for the management of fire extinctions (Castillo et al., 2006), for the planning of space missions (Bresina et al., 2005), etc . But despite of all these successful examples, the application of AP systems to real-world problems is still complicated, mainly due to the following two factors. First, the search for a solution plan in AP is a PSpace-complete¹ problem (Bylander, 1991). One can easily find planning problems that overwhelm the capacities of the off-the-shelf planners. Second, AI planners need to be fed with accurate description of the planning tasks. These descriptions consist of a model of the actions that can be carried out in the environment together with a specification of the state of the environment and the goals to achieve. Knowing in advance this information is unfeasible in most of the real-world problems.

The following sections describe different approaches for applying Machine Learning (ML) to assist AP in order to overcome these two problems. These approaches learn either heuristics (also called control knowledge) for handling the search complexity or action models. The chapter is organized as follows. The first section explains the basic concepts of AP. Second section describes the common issues about using ML for AP. Third section reviews the last techniques for automatically learning AP control knowledge. Fourth section reviews the last advances for the learning of AP action models. The fifth section depicts the current challenges of ML for AP. And finally, section sixth discusses some conclusions.

BACKGROUND

An AP task is defined by two elements: (1) a set of actions that represents the state-transition function of the world (*the planning domain*) and (2), a set of facts that represent the initial state together with the goals of the AP task (*the planning problem*). These two elements are typically represented in languages coming from the first-order logic. In the early days of AP, STRIPS was the most popular representation language. In 1998 the Planning Domain Definition Language (PDDL) was developed for the first International Planning Competition (IPC). Since that date, PDDL has become the standard representation language for the AP community. According to the PDDL specification (Fox & Long, 2003), an action in the planning domain is represented by: (1) the action preconditions, a list of predicates indicating the facts that must be true so the action becomes applicable and (2) the action effects, which is a list of predicates indicating the changes in the state after the action application. Like STRIPS, PDDL follows the closed world assumption to solve the frame problem. Regarding this assumption, what is not currently known to be true, is false.

Before the mid ‘90s, automated planners could only synthesize plans of no more than 10 actions in an acceptable amount of time. During those years, planners strongly depended on speedup techniques, such as learning control knowledge, for solving interesting AP problems. In the late 90’s, a significant scale-up in planning took place due to the appearance of the reachability planning graph (Blum & Furst, 1995) and the development of powerful domain independent heuristics (Hoffman & Nebel, 2001; Bonet & Geffner, 2001). Planners using these advances can often synthesize 100-action plans just in seconds. So, at the present time, there is not such dependence on ML for solving AP problems. However, there is a renewed interest in learning for AP motivated by two factors: (1) IPC-2000 showed that knowledge-based planners scale up dramatically better than domain-independent planners. The development of ML techniques that automatically define the kind of knowledge that humans put in these planners would bring great advances to the field. With this aim, the IPC-2008 introduced a specific track for learning-

based planners. And, (2) there is a need for tools that assist in the definition, validation and maintenance of the AP models. At the moment, these processes are still done by hand. Since 2005, the International Competition on Knowledge Engineering for Planning Systems (ICKEPS) takes place every two years with the aim of encouraging the development of tools for modeling AP tasks.

LEARNING ISSUES WITHIN AUTOMATED PLANNING

Some common issues has to be considered when designing a learning process for the assistance of a given AP system. The most relevant ones are: (1) How to represent the learned knowledge, (2) which algorithm is suitable for learning this knowledge, and (3) how to collect examples for learning this knowledge. Though there are no general answers, this section gives clues to these questions.

The Representation Language

AP tasks are commonly defined in predicate logic because it allows one to represent problems in a compact way and to capture relations between objects. However, other representation languages have been used to express the learned knowledge in order to make the learning processes more efficient. Languages for describing object classes has been shown to provide a useful learning bias for many AP domains. These languages include *Concept Language* (Martin & Geffner, 2000) or *Taxonomic Syntax* (McAllester & Givan, 1993). These encoding have the expressive power of standard first order logic with a syntax that is suited for representing and reasoning with classes of objects. For example, the useful concept of block “well-placed” in the Blocksworld domain² can be defined using these languages in a very compact way: *A block is well placed when it is on the right block or table and all blocks beneath it are “well-placed” too.*

When the AP task is compiled into another form of problem solving, e.g., Constraint Satisfaction Problems (CSP), Boolean Satisfiability Problems (SAT) or Model Checking, the representation language of the learned knowledge must suit to the corresponding compilation. In the case of Model Checking planners, like the TLPlan system (Bacchus & Ady, 2001), control knowledge is represented as new plan requirements expressed by temporal logic formulas. In the case of planners based on CSP, like the system GP-CSP (Do & Kambhampati, 2000), search failures were stored as variable-value assignations with the meaning of not being part of the solution.

The selection of a *feature space* able to capture the significant knowledge of the domain is also a key issue. Traditionally, this *feature space* has consisted on predicates for describing the *current state* and the *goals* of the planning task. The PRODIGY system (Minton 1988) enriched the *feature space* with extra predicates, called meta-predicates (Borrado & Veloso, 1997). Meta-predicates captured useful knowledge of the planning context, like the current applicable operators or the goals pending to be solved. Recent works on learning general policies has extended this definition of meta-predicates including predicates for capturing the actions of the relaxed plan in the current state (Yoon et al., 2007) or the current helpful actions (de la Rosa et al., 2008).

The Learning Algorithms

One of the most significant issues that arises in applying ML to AP is the learning algorithm. Next, there is a description of the ML mechanisms that best suit the AP task.

Inductive Logic Programming

Inductive Logic Programming (ILP) arises from the intersection of ML and logic programming and is concerned about the development of inductive algorithms to learn a given target logic concept, from examples and background knowledge. Formally, the inputs to an ILP system are:

- A set of ground learning examples E of the target concept, consisting of true examples (positive examples) E^+ and false examples (negative examples) E^- of the concept.
- A representation language L , specifying the syntactic restrictions of the domain predicates.
- Background knowledge B , which provides additional information to argument the classification of the learning examples

With these three inputs, the ILP systems try to find a hypothesis H , expressed in L which agrees with the examples E and the background knowledge B .

ILP has traditionally been considered a binary classification task for the given target concept. However, in the recent years, ILP techniques have broadened to cover the whole spectrum of ML tasks such as regression, clustering or association analysis. Particularly, there is a successful new trend in ILP consisting on extending the classical propositional ML algorithms to the relational framework:

- Learning relational decision trees. A very well-known approach for multi-class classification consists of finding the smallest decision tree that fits a given data set. The common way to find these decision trees is following the *Top-Down Induction of Decision Trees* (TDIDT) algorithm (Quinlan, 1986). This algorithm builds the decision tree by splitting the training examples according to the values of a selected attribute that minimizes a measure of variance along the prediction variable. The leaves of learned trees are labeled by a class value for the target concept that fit the examples that satisfy the conditions along the path from the root of the tree to that leaf. Relational decision trees are the first-order logic upgrade of the classical decision trees (Blockeel & De Raedt, 1998). Unlike the classical ones, relational trees work with examples described in a relational language such as predicate logic. This means that each example is not described by a single feature vector but by a set of logic facts. Therefore, the nodes of the tree do not contain tests about the examples attributes, but logic queries about the facts holding in the examples.
- Relational instance based learning. Unlike tree learning methods that explicitly construct a classifier from learning examples, these techniques perform what is called a *lazy learning*, i.e. they simply store the learning examples and delay the generalization until the moment a new example have to be classified. One disadvantage of these methods is that the cost of classifying an instance can be high because it implies all the generalization computations. On the other hand the generalization of these methods is local to the neighborhood of the new example which allows one to achieve better performance when the classification function is very complex. The generalization in all these methods is done by analyzing instances similar to the new query instance ignoring

the ones that are very different. Thereby a critical issue is the definition of an accurate similarity measure among the instances described in predicate logic (Sebag, 1997; Ramon & Bruynooghe, 2001).

Explanation Based Learning

EBL explains the learning examples using the domain theory. An explanation is a justification for why a subset of learning examples merits its assigned class. Other learning examples are used to evaluate these conjectured explanations. The informational effect of a confirmed explanation is equivalent to all examples covered by the explanation. Since this can be significantly larger than the learning set, the learner can draw much stronger statistical conclusions than would be warranted from the learning examples alone. Formally, the inputs to an EBL system are:

- A set of learning examples E .
- A domain theory D consisting of correct predicate definitions.
- The current hypothesis H which is incorrect or should be redefined in other terms.
- A representation language L specifying the syntactic constraints on the predicate definitions.

The EBL task is formally defined as learning a new hypothesis H' such as H' is complete and consistent with the learning examples and the domain theory D .

Reinforcement Learning

ILP and EBL shared a common feature: the learning success depended on outside expertise (1) to decide what is worth to learn, i.e., the learning target and (2) to collect significant examples of the learning target e.g., the set of positive and negative examples in ILP for learning a given logic concept. But such outside expertise is not always available. Reinforcement Learning (RL) poses a different approach to deal with this limitation: RL tries to learn how to take a decision in a given environment by trial and error. According to this, RL reduces the external supervision to a reward signal guidance.

Most of the work on RL has focused on propositional representations for the state and the actions. Because of that, traditional RL algorithms are not applicable to domains with relational structure, like the AP ones, without extensive engineering effort. Recently, there are coming up numerous attempts to naturally bring together RL and the expressiveness of relational representations for states and actions. They are included in a new AI field called Relational Reinforcement Learning (RRL). Currently, the different approaches of RRL to generalize RL to the relational setting consist of:

- Relational learning of the value function. This approach consists of using relational regression tools to generalize the value function. So far, three different relational regression algorithms have been used:
 - Relational regression trees (Dzeroski et al., 2001). For each goal, a regression tree is build from examples consisted on pairs (state,q-value). The test nodes of the induced tree represent the set of facts that have to be holding for the prediction to be true. The leaf nodes of tree represent the different predictions of the q-value.

- Instance based algorithm with relational distance (Driessens & Ramon, 2003). In this case a k-nearest neighbor prediction is done. It computes a weighted average of the Q-values of the examples stored in memory where the weight is inversely proportional to the distance between the examples. This distance measure copes with the relational representations of the states and actions of the examples.
- Relational Kernels methods (Gartner et al., 2003). They use the incremental regression algorithm *Gaussian Processes* to approximate the mappings between q-values and state-action pairs. In order to employ *Gaussian Processes* in a relational setting they make use graph kernels as the covariance function between state-action pairs.
- Relational learning of policies. An important drawback of the previous methods is that the value function can be very hard to predict in stochastic tasks. An alternative approach is trying to directly learn the policies and only implicitly represent the value function. That is, given an explicit representation of a policy pi , the implicitly represented value function is the value obtained by executing pi repeatedly from each state. Note that in this case, a classifier, like relational decision trees (Dzeroski et al., 2001), is needed to learn the optimal pi rather than the regression learners previously used. The advantage of this alternative approach is that usually it is easier to represent and learn suitable policies for structured domains than to represent and learn accurate value functions.

The Generation of Learning Examples

The success of the ML algorithms strongly depends on the learning examples used. In the case of AP, these learning examples are extracted from the experience collected solving training problems. Therefore, the quality of the learning examples will depend on the quality of the problems used for training. Traditionally, these training problems are obtained by random generators with some parameters to tune the problems difficulty. This approach presents two limitations. First, is not trivial to guarantee that a given AP problem is solvable. Formally, proving that from a given initial state, one can reach a state where goals are true is as hard as the original AP task. Second, the parameters of the AP problem generators are domain-dependent (the number and type of the objects of the world are different for different AP domains). Tuning these parameters for generating good quality learning examples implies domain expertise.

In order to avoid the domain expertise for obtaining good learning examples, (Yoon et al., 2004) introduced *Random Walks*, which is an automatic and domain-independent strategy for “bootstrapping” the learning process by using automatically generated problems of gradually increasing walk length. First, it generates a random initial state s_0 . Second, starting at s_0 it takes n uniformly random actions (only select random actions from the set of applicable actions in a state s_i) to produce the state sequence (s_0, s_1, \dots, s_n) . Finally, it returns the planning problem with initial state s_0 and the set of goals s_n . This approach only requires the availability of an action simulator for the planning domain. However, in domains where complexity depends on the number of world objects this approach is not enough.

An alternative approach consists of generating training problems using a previous one (Fuentetaja & Borrajo, 2006). In this case the learning process has the bias imposed by this previous problem, but it allows generating a random exploration of problems of increasing difficulty in domains where complexity depends on the number of world objects. This approach also presents a limitation. When the domain is not symmetrical (i.e. for each instantiated operator o_i), there must be a sequence of instantiated opera-

Figure 1. Example of macro-action induced by MacroFF for the Depots domain.

```
(:action UNLOAD
  :parameters (?x - hoist ?y - crate ?t - truck ?p - place)
  :precondition (and (in ?y ?t) (available ?x) (at ?t ?p) (at ?x ?p))
  :effect (and (not (in ?y ?t)) (not (available ?x)) (lifting ?x ?y)))

(:action DROP
  :parameters (?x - hoist ?y - crate ?s - surface ?p - place)
  :precondition (and (lifting ?x ?y) (clear ?s) (at ?s ?p) (at ?x ?p))
  :effect (and (available ?x) (not (lifting ?x ?y)) (at ?y ?p)
    (not (clear ?s)) (clear ?y) (on ?y ?s)))

(:action UNLOAD-DROP
  :parameters (?h - hoist ?c - crate ?t - truck ?p - place ?s - surface)
  :precondition (and (at ?h ?p) (in ?c ?t) (available ?h)
    (at ?t ?p) (clear ?s) (at ?s ?p))
  :effect (and (not (in ?c ?t)) (not (clear ?s))
    (at ?c ?p) (clear ?c) (on ?c ?s)))
```

tors, such that, if applied, they return to the planning state before applying o_i , otherwise, it could easily generate unsolvable problems.

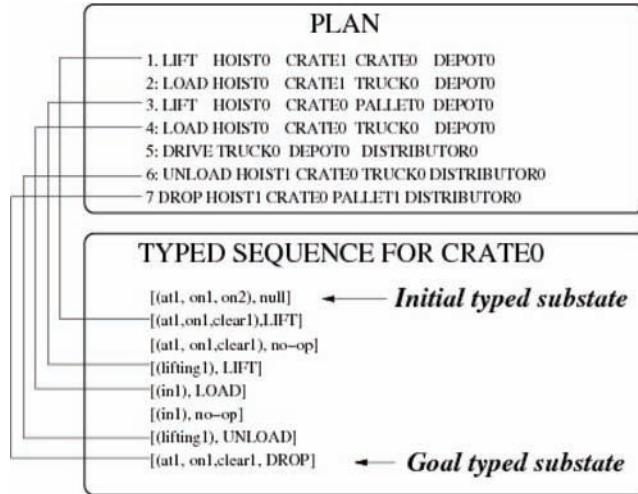
LEARNING PLANNING CONTROL KNOWLEDGE

On one hand, the planning systems that are able to exploit domain-specific knowledge can perform orders of magnitude faster than the off-the-shelf planners. On the other, real-world applications make use of control knowledge to find solution plans fitting the user preferences. In both cases, hand-coding this domain-specific control knowledge is tricky because it implies expertise on of both the planning domain and the planning system. This section revises the different techniques for automatically learning search control knowledge according to the target of their learning process: planning sequences, control rules, planning policies, hierarchies or evaluation function.

Learning Planning Sequences

The target of learning sequences refers to capture portions of planning episodes that can occur in other problems; therefore a previous recorded situation may help in the process of solving the new problem. If the sequences becomes directly from solution plans they are called macro-actions. If the sequences are abstractions of decisions taken during the planning search, they can take different means, but generally called planning cases.

Figure 2. Example of a planning case learned for the Depots domain.



Learning MacroActions

Macro-actions are the first attempt to speed-up the planning process. They are extra actions added to a given domain theory resulting from combining the actions that are frequently used together. Figure 1 shows an example of the macro-action UNLOAD-DROP induced by the MacroFF (Botea et al., 2005) system for the Depots domain³.

The use of macro-actions reduces the depth of the search tree. However, this benefit decreases with the number of new macro-actions as they enlarge the branching factor of the search tree causing the utility problem. To decrease this negative effect filters deciding the applicability of the macro-actions should be defined.

Since the beginning of AP the use of macro-actions has been widely explored. Traditionally, most of the techniques use an off-line approach to generate and filter macro-actions before using them in search. Early works on macro-actions began with a version of the STRIPS planner (Fikes et al., 1972). It used previous solution plans and segments of the plans as macro-actions to solve subsequent problems. Later, (Korf, 1985) extended this approach by adding some filtering heuristics to prune the generated set of macro-actions. This approach distinguished between two types of macro-actions: S-macros that occur frequently during search and T-macros, those that occur less often, but model some weakness in the heuristic. The REFLECT system took the alternative approach of forming macro-actions based on pre-processing of the domain (Dawson & Silklossy, 1977). All sound pairwise combinations of actions were considered as macro-actions and filtered through some basic pruning rules. Due to the small size of the domains with which the planner was reasoning, the number of macro-actions remaining following this process was sufficiently small to use in planning.

More recent works on macro-actions include the IPC4 competitor MacroFF (Botea et al., 2005). In this work macro-actions are extracted in two ways: from solution plans and by the identification of statically connected abstract components. Besides, an off-line filtering technique is used to prune the list of macro-actions. Marvin also competed in IPC4 using action-sequence-memorization techniques to generate macro-actions on-line that allow the planner escape from plateaus without any exploration

Figure 3. Example of a control rule learned by PRODIGY for the Depots domain.

```
(control-rule select-load-for-available-hoist
  (if (and (current-goal (available <h1>))
            (type-of-object <c1> crate)
            (type-of-object <h1> hoist)
            (type-of-object <l1> place)
            (type-of-object <s1> surface)))
      (then select operator load))
```

(Coles & Smith, 2007). Recently, (Newton, 2007) uses a genetic algorithm to generate a collection of macro-actions independently of the source planner, and then filters this collection through an off-line filtering technique similar to that used by MacroFF.

Learning Planning Cases

Planning Cases (an example shown in Figure 2) consist of traces of past solved planning problems. Unlike macro-actions, cases can memorize goals information. PRODIGY/ANALOGY (Veloso & Carbonell, 1993) introduced the application of derivational analogy and abstraction to planning. This system stored planning traces to avoid failure paths in future exploration of the search tree. To retrieve similar planning traces, PRODIGY/ANALOGY indexed them using the minimum preconditions to reach a set of goals. Recently, typed sequences cases have been used in heuristic planning for node ordering during the plan search (de la Rosa et al., 2007). They have shown that a reduction in the number of heuristic evaluations can improve the planner performance in terms of time. Contrary to macro-actions, this technique is object-centered, useful for domains in which different goal situations determine the plan construction.

Learning Control Rules

A control rule is an *IF-THEN* rule for guiding the exploration of the planning search tree. Control rules can guide the exploration in two different ways: for node pruning or for node ordering during the tree exploration. Figure 3 shows and example of a control rule for node punning learned for the PRODIGY (Minton 1988) system in the Depots domain.

Control rules enrich the planning language with extra predicates, called meta-predicates. Meta-predicates allow the planner to capture specific knowledge of the given domain, e.g., the applicable operator, the current goals the planner is trying to solve. However, the knowledge captured depends strongly on the quality of the learning examples used. When the learning examples are not significant enough the induced control rules may mislead the search process of the planner. Besides, control rules also suffer from the utility problem.

On one hand, there are systems that inductively learn control rules. Among these systems Inductive Learning Programming (ILP) is the most popular learning technique. The Grasshopper system (Leckie & Zukerman, 1991) used the Foil (Quinlan & Cameron-Jones, 1995) ILP system to learn control rules that guide the PRODIGY planner when selecting goals, operators and binding operators. Purely inductive systems need a big number of learning examples to acquire efficient control knowledge. On the

other hand, there are analytical systems. Static that obtains control rules without any training example just using EBL to analyze the relations between actions' preconditions and effects (Etzioni, 1993). The main disadvantage of these methods is that, as there are no training examples, there is no measure of the learned knowledge utility. With the aim of solving the problems of the purely inductive and purely analytical approaches some researchers have tried to combine them: the pioneering systems based on this principle are LEX-2 (Mitchell et al., 1982) and Meta-Lex (Keller, 1987). AxA-EBL combined EBL + induction (Cohen, 1990). It first learns control rules with EBG and then refines them with training examples. Dolphin was an extension of AxA-EBL which used Foil as the inductive learning module (Zelle & Mooney, 1993). The Hamlet system combines deduction and induction in a incremental way (Borrado & Veloso, 1997). First it learns control rules with EBL that usually are too specific and then uses induction to generalize and correct the rules. EvoCK applies the genetic programming paradigm to solve the learning problems caused by the hypothesis representation language (Aler et al., 2002).

Learning General Policies

A policy is a mapping between the world states and the preferred action to be executed in order to achieve a certain set of goals. A general policy it is a mapping for the problem instances of a given domain, i.e. the diverse combinations of initial state and goals, into the preferred action to be executed in order to achieve the goals. Thereby, a good general policy is able to solve all the possible problems instances of a given domain. However, though no general method is known for computing such general policies, they can be defined by-hand for many domains. For example, a general policy for the Blocksworld domain can be given as follows:

- (1) put all blocks on the table, and then
- (2) move block x on block y when x should be on y,
both blocks are clear, and y is well placed

The problem of learning general policies was first studied in (Kharden, 1999). Kharden induced general policies for both the Blocksworld and the Logistics domain by extending the decision list learning algorithm (Rivest, 1987) to the relational setting. This first approach presented two important weaknesses: (1) it relied on human-defined background knowledge that expressed key features of the domain, e.g. the predicates *above(block1,block2)* or *in_place(block)* for the Blocksworld, and (2) the learned policies do not generalize well when the problem size is increased. Martin and Geffner solved the weaknesses of Kharden's approach in the Blocksworld domain by changing the representation language of the general policies (Martin & Geffner, 2000). Instead of representing the current state and the problems goals using predicate logics they used a Concept Language. This representation allows them to learn rules of the form:

IF an object is in a certain class,
THEN do a certain action on the object.

In the last years the scope of general policy learning has been augmented over a diversity of domains making this approach to be competitive with the state-of-the-art planners. This achievement is due to the assimilation of two new ideas: (1) The policy representation language is enriched with new predicates.

Figure 4. Example of a method for hierarchical planning in the Depots domain.

```
(:method
  ; head
    (transport-crate ?crate ?destination)
  ; precondition
    (and
      (truck ?truck)
      (at ?truck ?place1)
      (at ?crate ?place2)
      (different ?place1 ?place2)))
  ; subtasks
    (:ordered (drive ?truck ?place1)
              (load ?crate ?truck)
              (drive ?truck ?place2)
              (unload ?crate ?truck)))
```

Meta-predicates are introduced to capture more effective domain specific knowledge. (2) The learned policies are not longer greedily applied but combined with heuristic planning algorithms. Specifically (Yoon et al., 2007) compliments the information of the current state with the relaxed planning graph and the learned policies are used during node expansions in a best-first search heuristic algorithm. At each node expansion of the best-first search, they add to the search queue the successors of the current best node as usual, but they also add the states encountered by following the policy from the current node for a given horizon.

Recently, the system Roller has defined the problem of learning general policies as a two-step standard classification process (de la Rosa et al., 2008). As a consequence, an off-the-shelf relational classifier can be used for general policy learning. At the first step the classifier captures the preferred operator to be applied in the different planning contexts. At the second step the classifier captures the preferred instantiation for each operator in the different planning contexts of a given domain. These contexts are defined by the set of helpful actions extracted from the relaxed planning graph of a given state, the goals remaining to be achieved, and the static predicates of the planning task. Additionally, Roller implemented two methods for guiding the search of a heuristic planner with the learned policies. The first one consists of using the resulting the policy in a Depth First Search algorithm. The second one consists of ordering the node evaluation of the Enforced Hill Climbing algorithm with the learned policy.

Learning Hierarchical Knowledge

Hierarchical planning combines hierarchical domain-specific representation of the planning models together with domain-independent search for problem solving. One of the best-known approaches for modeling hierarchical knowledge about a planning domain is Hierarchical Task Network (HTN). Current HTN planners can outperform the state-of-the-art ones and provide a natural modeling framework for

many real-world applications like fire extinctions (Castillo et al., 2006), evacuation planning (Muñoz-Avila et al., 1999) or manufacturing.

In HTN planning, complex tasks are decomposed into simpler tasks until a sequence of primitive actions is generated. Therefore, the input to an HTN planner includes a set of operators similar to the ones used in classical planning (primitive actions) and a set of methods describing how tasks should be decomposed into subtasks in this particular domain. Figure 4 shows the method for the task *transport-crate(crate,place)* from HTN description of the Depots domain.

The specification of these hierarchical methods by hand is a hard task as expert knowledge is needed. And defining a general algorithm for automatically learning them for any domain is still an open issue. The ALPINE system completely automates the generation of abstraction hierarchies from the definition of a problem space (Knoblock, 1990). Each abstraction space in a hierarchy is formed by dropping literals from the original problem space, thus it abstracts the preconditions and effects of operators as well as the states and goals of a problem. Concerning abstraction in planning, there also exists the system PARIS that stores each case in different levels of abstraction (Bergmann & Wilke, 1992). To solve new problems, the problem will be compared with cases of the hierarchy of abstractions and the planner is used to refine the abstract case and to adapt it to the new problem. X-learn is a system that uses a generalize-and-test algorithm based on ILP to learn goal-decomposition rules (Reddy & Tadepalli, 1997). These (potentially recursive) rules are 3-tuples that tell the planner how to decompose a goal into a sequence of subgoals in a given world state, and therefore are functionally similar to methods in HTN domains. X-learn training data consists of solutions to the planning problems ordered in an increasing order of difficulty (authors refer to this training set as an exercise set, as opposed to an example set, which is a set of random training samples without any particular order). This simple-to-hard order in the training set is based on the observation that simple planning problems are often subproblems of harder problems and therefore learning how to solve simpler problems will potentially be useful in solving hard ones. Hicap (Hierarchical Interactive Case-based Architecture for Planning) integrates the SHOP hierarchical planner together with a case-based reasoning (CBR) system named NaCoDAE to assist with the authoring of plans for noncombatant evacuation operations (Muñoz-Avila et al., 1999). It interacts with users to extract a stored similar case that allows one to solve the current problem. The system CaseAdvisor also integrates CBR and hierarchical planning (Carrick et al., 1999). It uses plans previously stored to obtain information of how to solve a task instead of choosing the refining method. The KnoMic (Knowledge Mimic) is a ML system which extracts hierarchical performance knowledge by observation to develop automated agents that intelligently perform complex real world tasks (van Lent & Laird, 2001). The knowledge representation learned by KnoMic is a specialized form of Soar's production rule format (Rosenbloom et al., 1993). The rules implement a hierarchy of operators with higher level operators having multiple sub-operators representing steps in the completion of the high level operator. These rules are then used by an agent to perform the same task. Langley and Rogers describes how ICARUS, a cognitive architecture that stores its knowledge of the world in two hierarchical categories of concept memory and skill memory, can learn these hierarchies by observing problem solving in sample domains (Langley & Choi, 2006).

Recent works attempt to automatically learn HTN domain descriptions: CaMeL uses an extended version of the Candidate Elimination incremental algorithm to learn expansions methods in a HTN planner by observing plan traces. It is designed for domains in which the planner is given multiple methods per task, but not their preconditions. That is, the structure of the hierarchy is known in advance and the

learning task is to identify under what conditions different hierarchies are applicable. The problem with this work is that it required very many plan traces to converge (completely determine the preconditions of all methods). CaMeL++ (Ilghami et al., 2005) is also an algorithm for learning preconditions for HTN methods that enables the planner to start planning before the method preconditions are fully learned. By doing so, the planner can start solving planning problems with a smaller number of training examples than is required to learn the preconditions completely with insignificant cost of few incorrect plans. Camel required all information about the methods except for their preconditions to be given to the learner in advance, so that the only information for the learner to learn was the methods preconditions. Besides, each input plan trace for Camel needed to contain a lot of information so that the learner could learn from it: At each decomposition point in a plan trace, the learner needed to have all the applicable method instances, rather than just the one that was actually used. The HDL algorithm introduced in (Ilghami et al., 2006) starts with no prior information about the methods, HDL does not need most of that information. At each decomposition point, it only needs to know about one or at most two methods: The method that was actually used to decompose the corresponding task, and one (if there are any) of the methods that matched that task but whose preconditions failed (to serve as a negative training sample). The system HTN-MAKER receives even less input information (Hogg et al., 2008). HTN-Maker is able to produce an HTN domain model from a STRIPS domain model, a collection of STRIPS plans, and a collection of task definitions.

Apart from learning decomposition task methods, learning has also been applied to hierarchical planning for other purposes. Lotem and Nau build a method for extracting knowledge on HTN planning problems for speeding up the search (Lotem & Nau, 2000). This knowledge is gathered by propagating properties through an AND/OR tree that represents disjunctively all possible decompositions of an HTN planning problem. This knowledge is used during the search process of the GraphHTN planner, to split the current refined planning problem into independent subproblems.

Learning Heuristic Functions

A heuristic function $H(s;A;g)$ is a function of a state s , an action set A , and a goals set g that estimates the cost of achieving the goals g starting from s and using the actions from A . In case the estimation provided by the heuristic function is accurate, a greedy application of the actions recommended by the heuristic will achieve the goals without search. However, when a heuristic is imperfect, it must be used in the context of a heuristic search algorithm, where the accuracy of the heuristic impacts the search efficiency.

The current domain-independent heuristics for AP are very expensive to compute. As a consequence, heuristic planners suffer from scalability problems. This effect becomes more prominent in domains where the heuristic function is less accurate because, in these domains heuristic planners compute useless node evaluations, e.g. Blocksworld domain. With the aim of avoiding this undesirable effect one can try to directly learn the heuristic function for a given domain: Regarding this approach, (Yoon et al., 2006; Xu et al., 2007) build a state generalized heuristic function through a regression process. The regression examples consist of observations of the true distance to the goal from diverse states, together with extra information from the relaxed planning graph. The obtained heuristic function provide a more accurate estimation that captures domain specific regularities expressed as a weighted linear combinations of features, that is,

$$H(s;A;g) = w_i * f_i(s;A;g),$$

where the w_i are the weights and the f_i represent the different features of the planning context. The main disadvantage of this leaning approach is that the result of the regression is poorly understandable by humans making the verification of the correctness of the acquired knowledge difficult.

LEARNING PLANNING ACTION MODELS

This section reviews the planning systems that use ML techniques for automatically acquiring knowledge about domain models. Because AP is a model-based form of problem solving it requires complete and correct definition of actions models. However, building action models from scratch is a difficult and time-consuming task, even for planning experts. The following ML techniques learn the parameters, preconditions and/or the effects of planning actions for the automatic definition of actions' models.

The LIVE system is an extension of the GPS problem solving framework with a learning component for rule learning (Shen & Simon, 1989). LIVE alternates problem solving with the rule learning for the automatic definition of STRIPS-like operators while exploring the world. The decision for alternation mainly depends on *surprises*, i.e., situations where an action's consequences violate its predicted models. When no rule can be found for solving the problem, LIVE will generate and execute an exploration plan, or a sequence of actions, seeking for surprises to extend the rule set. When new rules are learned, problem solving is resumed and a solution plan may be constructed through means-ends analysis. Since the correctness of a solution plan cannot be guaranteed, learning is inevitable at execution time. When the prediction of a rule fails during execution, LIVE will revise the rule set and then plan another solution for the problem.

The EXPO system refines incomplete planning operators (are missing some preconditions and effects) when the monitoring of a plan execution detects a divergence between internal expectations and external observations (Gil, 1997). Plans are executed while the external environment is monitored, and differences between the internal state and external observations are detected by various methods each correlated with a typical cause for the expectation failure. The methods also construct a set of concrete hypotheses to repair the knowledge deficit. After being heuristically filtered, each hypothesis is tested in turn with an experiment. After the experiment is designed, a plan is constructed to achieve the situation required to carry out the experiment. The experiment plan must meet constraints such as minimizing plan length and negative interference with the main goals.

Unlike the previous works that refined planning operators by an active exploration of the environment, OBSERVER learns PRODIGY operators by observing expert agents and subsequent knowledge refinement in a learning-by-doing paradigm (Wang, 1994). The observations of the expert agent consist of: (1) the sequence of actions being executed, (2) the state in which each action is executed, and (3) the state resulting from the execution of each action. Planning operators are learned from these observation sequences in an incremental fashion utilizing a conservative specific-to-general inductive generalization process. In order to refine the new operators to make them correct and complete, the system uses the new operators to solve practice problems, analyzing and learning from the execution traces of the resulting solutions or execution failures.

The TRAIL system limits its operators to an extended version of Horn clauses so that ILP can be applied (Benson, 1997) . TRAIL learns the *preimage* or precondition of an action for a TOP operator using ILP. The examples used require the positive or negative examples of propositions held in states just before each actions application. This enables a concept for the *preimage* of an action to be learned for each state just before that action. TRAIL learned well when the positive and negative examples of states before all target actions are given.

The LOPE system achieves learning planning operators by observing the consequences of executing planned actions in the environment (Garcia-Martinez & Borrajo, 2000) . At the beginning, the system has no knowledge, it perceives the initial situation, and selects a random action to execute in the environment. Then it loops by (1) executing an action, (2) perceiving the resulting situation of the action execution and its utility, (3) learning a model from the perception and (4) planning for further interaction with the environment in case the execution of the plan is finished, or when the system observes a mismatch between the predicted situation and the situation perceived. The planning component of LOPE does not explicitly receive a goal input given that LOPE creates its own goals from the situations with the highest utility.

Besides all these systems, the techniques previously revised for the automatic definition of decomposition methods for HTN planning are also considered as learning action's schema techniques

In domains with uncertainty less work has been done: (Pasula et al., 2007) presented the first specific algorithm for learning probabilistic planning operators. (Jiménez et al., 2008) introduced a mechanism for automatically complete a STRIPS action model with situation-dependent probabilities learned from plan executions. The ARMS system learns preconditions and effects of domain operators only from initial state, goal state and plan in domains where no or partial intermediate states are given (Yang et al., 2005). (Amir, 2006) introduced an algorithm to exactly learn actions in partially observable STRIPS domains. But still, there is no general efficient approach yet to cope with the problem of partial observability and stochastic environment.

FUTURE TRENDS

Since the STRIPS days, the planning representation languages have evolved to bring together AP algorithms and real-world problems. Nowadays, the PDDL version for the IPC-2008 includes numeric state variables to support quality metrics, durative actions that allow explicit time representation, derived predicates to enrich the descriptions of the system states, and soft goals and trajectory constraints to express user preferences about the different possible plans. Nevertheless, most of these new features are not efficiently handled by the state-of-the-art planners: they add such extra complexity to the search process that problems become extremely difficult to solve. Besides, off-the-shelf planners still fail to scale-up in domains with strong goals interaction, such as the Blocksworld. As it is very difficult to find an efficient general solution to all these challenges, ML must play an important role in addressing them because it can alleviates the complexity of the search process by exploiting regularity in the space of common problems.

Additionally, the state-of-the-art planning algorithms need a detailed domain description to efficiently solve the AP tasks, but real-world applications like controlling autonomous vehicles, emergency evacuations, etc, imply planning in environments where the dynamics model may be not easily accessible. There is a current need for planning systems capable of progressively acquiring and refining planning

models of the environment as more information is available. This issue has already been extensively studied in other AI areas such as RL.

CONCLUSION

We have described in this chapter the state-of-the-art in ML for assisting AI planning. Automatic learned knowledge is useful for AP in two different ways: (1) it helps planners guiding their search processes and (2), it assists planning domain designers on the definition of planning action models.

As shown in the paper there is a whole bunch of ML techniques that successfully help planning to solve particular problems. However, research in learning based planning is not mature from a general problem solving point of view. Unlike off-the-shelf automated planners, the learning based systems are usually not able to obtain good performance in a diversity of domains, they frequently implement ad-hoc learning algorithms so they can not benefit from the last advances in ML, and moreover, they lack of theoretical mechanisms to evaluate the utility of the learning knowledge. The specific IPC track for learning control knowledge together with the ICCEPS, promise to settle the formal basis for learning-based AP. This basis involve finding methodologies for system comparisons, building frameworks for rapid algorithm development and test, and keeping the community interest on new challenging problems.

REFERENCES

- Aler, R., Borrajo, D., & Isasi, P. (2002). Using genetic programming to learn and improve control knowledge. *Artificial Intelligence*, 141, 29–56. doi:10.1016/S0004-3702(02)00246-1
- Amir, E. (2006). Learning partially observable action schemas. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence*.
- Bacchus, F., & Ady, M. (2001). Planning with resources and concurrency: A forward chaining approach. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence* (pp. 417-424).
- Bacchus, F., & Kabanza, F. (2000). Using temporal logics to express search control knowledge for planning. *Artificial Intelligence Journal*, 116, 123–191. doi:10.1016/S0004-3702(99)00071-5
- Benson, S. (1997). *Learning action models for reactive autonomous agents*. Unpublished doctoral dissertation, Stanford University.
- Bergmann, R., & Wilke, W. (1996). PARIS: Flexible plan adaptation by abstraction and refinement. In *Proceedings of the Workshop on Adaptation in Case-Based Reasoning, ECAI06*.
- Blockeel, H., & De Raedt, L. (1998). Top-down induction of first-order logical decision trees. *Artificial Intelligence*, 101, 285–297. doi:10.1016/S0004-3702(98)00034-4
- Blum, A. L., & Furst, M. L. (1995). Fast planning through planning graph analysis. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada (pp. 1636-1642).
- Bonet, B., & Geffner, H. (2001). Planning as heuristic search. *Artificial Intelligence*, 129, 5–33. doi:10.1016/S0004-3702(01)00108-4

- Borrajo, D., & Veloso, M. (1997). Lazy incremental learning of control knowledge for efficiently obtaining quality plans. *AI Review Journal*, 11, 371–340.
- Botea, A., Enzenberger, M., Muller, M., & Schaeffer, J. (2004). Macro-FF: Improving AI planning with automatically learned macro-operators. *Journal of Artificial Intelligence Research*, 24, 581–621.
- Bresina, J., Jonsson, A., Morris, P., & Rajan, K. (2005). Activity planning for the Mars exploration rovers. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- Bylander, T. (1991). Complexity results for planning. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, Sydney, Australia (pp. 274-279).
- Carrick, C., Yang, Q., Abi-zeid, I., & Lamontagne, L. (1999). Activating CBR systems through autonomous information gathering. In *Proceedings of the 3rd International Conference on Case-Based Reasoning*.
- Castillo, L., Fdez-Olivares, J., García-Pérez, O., & Palao, F. (2006). Bringing users and planning technology together. Experiences in SIADEX. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling*.
- Cohen, W. W. (1990). Learning approximate control rules of high utility. In *Proceedings of the International Conference on Machine Learning* (pp. 268-276).
- Coles, A., & Smith, A. (2007). Marvin: A heuristic search planner with online macro-action learning. *Journal of Artificial Intelligence Research*, 28, 119–156.
- Dawson, C., & Silklossy, L. (1977). The role of preprocessing in problem solving system. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence* (pp. 465-471).
- de la Rosa, T., Garcia-Olaya, A., & Borrajo, D. (2007). Using cases utility for heuristic planning improvement. In *Proceedings of the 7th International Conference on Case-Based Reasoning*, Belfast, Northern Ireland.
- de la Rosa, T., Jimenez, S., & Borrajo, D. (2008). Learning relational decision trees for guiding heuristic planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- Do, M. B., & Kambhampati, S. (2000). Solving planning graph by compiling it into a CSP. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems (AIPS00)*.
- Driessens, K., & Ramon, J. (2003). Relational instance based regression for relational reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.
- Dzeroski, S., De Raedt, L., & Driessens, K. (2001). Relational reinforcement learning. *Machine Learning*, 43, 7–52. doi:10.1023/A:1007694015589
- Etzioni, O. (1993). Acquiring search-control knowledge via static analysis. *Artificial Intelligence*, 62, 255–301. doi:10.1016/0004-3702(93)90080-U
- Fikes, R. E., Hart, P. E., & Nilsson, N. J. (1972). Learning and executing generalized robot plans. *Artificial Intelligence*, 3, 251–288. doi:10.1016/0004-3702(72)90051-3

- Fikes, R. E., & Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2, 189–208. doi:10.1016/0004-3702(71)90010-5
- Fox, M., & Long, D. (1998). The automatic inference of state invariants in TIM. *Journal of Artificial Intelligence Research*, 9, 367–421.
- Fox, M., & Long, D. (2002). Extending the exploitation of symmetries in planning. In *Proceedings of the International Conference on AI Planning and Scheduling*.
- Fox, M., & Long, D. (2003). PDDL2.1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20, 61–124.
- Fuentetaja, R., & Borrajo, D. (2006). Improving control-knowledge acquisition for planning by active learning. In *Proceedings of the European Conference on Machine Learning*.
- Garcia-Martinez, R., & Borrajo, D. (2000). An integrated approach of learning, planning, and execution. *Journal of Intelligent & Robotic Systems*, 29, 47–78. doi:10.1023/A:1008134010576
- Gartner, T., Driessens, K., & Ramon, J. (2003). Relational instance based regression for relational reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.
- Ghallab, M., Nau, D., & Traverso, P. (2004). *Automated task planning. Theory & practice*. San Francisco: Morgan Kaufmann.
- Gil, Y. (1992). *Acquiring domain knowledge for planning by experimentation*. Unpublished doctoral dissertation, Carnegie Mellon University.
- Hoffmann, J., & Nebel, B. (2001). The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14, 253–302.
- Hogg, C., Munoz-Avila, H., & Kuter, U. (2008). HTN-MAKER: Learning HTNs with Minimal Additional Knowledge Engineering Required. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*.
- Huffman, S. B., Pearson, D. J., & Laird, J. E. (1992). Correcting imperfect domain theories: A knowledge level analysis. In *Machine learning: Induction, analogy and discovery*.
- Ilghami, O., Nau, D., & Héctor Muñoz-Avila, H. (2006). Learning to do HTN planning. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling*.
- Ilghami, O., Nau, D., Muñoz-Avila, H., & Aha, D. W. (2002). CaMel: Learning method preconditions for HTN planning. In *Proceedings of the 6th International Conference on Artificial Intelligence Planning Systems (AIPS02)*.
- Jiménez, S., Fernández, F., & Borrajo, D. (2008). The PELA architecture: Integrating planning and learning to improve execution. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*.
- Keller, R. (1987). *The role of explicit contextual knowledge in learning concepts to improve performance* (Tech. Rep.). Rutgers University.

- Khardon, R. (1999). Learning action strategies for planning domains. *Artificial Intelligence*, 113, 125–148. doi:10.1016/S0004-3702(99)00060-0
- Knoblock, C. (1990). Learning abstraction hierarchies for problem solving. In *Proceedings of the Seventh International Workshop on Machine Learning*.
- Korf, R. E. (1985). Macro-operators: A weak method for learning. *Artificial Intelligence*, 26, 35–77. doi:10.1016/0004-3702(85)90012-8
- Langley, P., & Choi, D. (2006). A Unified cognitive architecture for physical agents. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence (AAAI'2006)*.
- Leckie, C., & Zukerman, I. (1991). Learning search control rules for planning: An inductive approach. In *Proceedings of the Eighth International Workshop on Machine Learning* (pp. 422-426).
- Lotem, A., & Nau, D. (2000). New advances in GraphHTN: Identifying independent subproblems in large HTN domains. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems* (pp. 206-215).
- Martin, M., & Geffner, H. (2000). Learning generalized policies in planning using concept languages. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems (AIPS00)*.
- Mcallester, D., & Givan, R. (1993). Taxonomic syntax for first order inference. *Journal of the ACM*, 40, 289–300.
- McGann, C., Py, F., Rajan, K., Ryan, H., & Henthorn, R. (2008). Adaptative control for autonomous underwater vehicles. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*.
- Minton, S. (1988). *Learning effective search control knowledge: an explanation based approach*. Amsterdam: Kluwer Academic Publisher.
- Mitchell, T., Utgoff, T., & Banerjiartin, R. (1982). Learning problem solving heuristics by experimentation. In *Machine learning: An artificial intelligence approach*.
- Muggleton, S. (1995). Inverse entailment and progol. *New Generation Computing*, 13, 245–286. doi:10.1007/BF03037227
- Muggleton, S., & Feng, C. (1990). Efficient induction of logic programs. In *Proceedings of the 1st Conference on Algorithmic Learning Theory* (pp. 368-381).
- Muñoz-Avila, H., Aha, D., Breslow, D., & Nau, D. (1999). HICAP: An interactive case based planning architecture and its application to non-combatant evacuation operations. In *Proceedings of the Ninth Conference on Innovative Applications of Artificial Intelligence*.
- Nau, D., Au, T., Ilghami, O., Kuter, U., Murdock, W., Wu, D., & Yaman, F. (2003). Shop: An HTN planning system. *Journal of Artificial Intelligence Research*, 20, 379–404.
- Nebel, B., Dimopoulos, Y., & Koehler, J. (1997). Ignoring irrelevant facts and operators in plan generation. In *Proceedings of the European conference on Planning*.

- Newton, M., Levine, J., Fox, M., & Long, D. (2007). Learning macro-actions for arbitrary planners and domains. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- Nilsson, N. J. (1984). *Shakey the robot* (Tech. Rep. 323). AI Center, SRI International, Menlo Park, CA.
- Pasula, H., Zettlemoyer, L., & Kaelbling, L. (2007). Learning symbolic models of stochastic domain. *Journal of Artificial Intelligence Research*, 29, 309–352.
- Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1, 81–106.
- Quinlan, J. R., & Cameron-Jones, R. M. (1995). Introduction of logic programs: {FOIL} and related systems. *New Generation Computing*, 13, 287–312. doi:10.1007/BF03037228
- Ramon, J., & Bruynooghe, M. (2001). A polynomial time computable metric between point sets. *Acta Informatica*, 37(10), 765–780. doi:10.1007/PL00013304
- Reddy, R., & Tadepalli, P. (1997). Learning goal-decomposition rules using exercises. In *Proceedings of the International Conference on Machine Learning* (pp. 278-286).
- Rivest, R. L. (1987). Learning Decision List. *Machine Learning*, 2, 229–246.
- Rosenbloom, P. S., Newell, A., & Lairdmon, J. E. (1993). Towards the knowledge level in Soar: The role of the architecture in the use of knowledge. *The Soar papers: Research on integrated intelligence*, 2, 897-933.
- Sevag, M. (1997). Distance induction in first order logic. In *Proceedings of the 7th International Workshop on Inductive Logic Programming* (pp. 264-272).
- Shen, W. M., & Simon, H. A. (1989). Rule creation and rule learning through environmental exploration. In *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 675-680).
- Sipser, M. (1997). *Introduction to the theory of computation*. Boston, MA: PWS Publishing.
- van Lent, M., & Laird, J. (2001). Learning procedural knowledge through observation. In *Proceedings of the International conference on Knowledge Capture*.
- Veloso, M., & Carbonell, J. (1993). Derivational analogy in PRODIGY: Automating case acquisition, storage, and utilization. *Machine Learning*, 10, 249–278. doi:10.1023/A:1022686910523
- Veloso, M., Carbonell, J., Pérez, A., Borrajo, D., Fink, E., & Blythe, J. (1995). Integrating planning and learning: The PRODIGY architecture. *JETAI*, 7(1), 81–120. doi:10.1080/09528139508953801
- Wang, X. (1994). Learning planning operators by observation and practice. In *Proceedings of the International Conference on AI Planning Systems* (pp. 335-340).
- Xu, Y., Fern, A., & Yoon, S. (2007). Discriminative learning of beam-search heuristics for planning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (pp. 2041-2046).
- Yang, Q., Wu, K., & Jiang, Y. (2005). Learning action models from plan examples with incomplete knowledge. In *Proceedings of the International Conference on Automated Planning and Scheduling*.

- Yoon, S., Fern, A., & Givan, R. (2004). Learning domain-specific control knowledge from random walks. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- Yoon, S., Fern, A., & Givan, R. (2006). Learning heuristic functions from relaxed plans. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- Yoon, S., Fern, A., & Givan, R. (2007). Using learned policies in heuristic-search planning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (pp. 1636-1642).
- Zelle, J., & Mooney, R. (1993). Combining FOIL and EBG to speed-up logic programs. In *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 1106-1113).

KEY TERMS AND DEFINITIONS

Bootstrap Effect: The fact of learning to solve a problem a teacher can not solve by observing how the teacher solves more simple problems.

Closed World Assumption: Any formula non explicitly asserted in a state is taken to be false, which allows one to avoid the explicit specification of negated literals. This assumption presupposes that actions only change a small part of the world.

Concept Language: also known as description logics, is a representation language with the expressive power of fragments of standard first-order logic but with a syntax that is suited for representing and reasoning with classes of objects.

The Frame Problem: The problem of expressing a dynamic system using logic without explicitly specifying which conditions are not affected by an action.

Meta-Predicates: are extra predicates used to reason about the state of the search process in the planner, e.g., the goals that the planner is currently working on, the operators being considered, etc.

Model Checking: consist of determining whether a given property, usually described as a temporal logic formula, holds in a given model of a system. Traditionally, this problem has been studied for the automatic verification of hardware circuits and network protocols.

Problem Distribution: A set of problems belonging to a given domain generated with the same number of world objects and problem goals.

Utility Problem: It is a drawback that arise when using learned knowledge cost of using overwhelms its benefit because the difficulty of storage and management the learned information and because of determining which information to use to solve a particular problem.

ENDNOTES

¹ PSpace-complete problems are those problems that are PSPACE (memory needs linear with respect to the length of the problem specification) and every problem in PSPACE can be reduced to it in polynomial time (Sipser, 1997).

² The Blocksworld is a classic domain in AP which consists of a set of blocks, a table and a gripper: the blocks can be on other blocks or on the table, a block that has nothing on it is clear and the gripper can hold one block or be empty. Because its simpleness and clarity it is by far the most

frequently domain used in the AI planning literature.

- ³ The Depots domain consists of actions to load and unload trucks, using hoists that are available at fixed locations. The loads are all crates that can be stacked and unstacked onto a fixed set of pallets at the locations.

Compilation of References

- Abbass, H. A., & Sarker, R. (2002). The Pareto differential evolution algorithm. *International Journal of Artificial Intelligence Tools*, 11(4), 531–552. doi:10.1142/S0218213002001039
- Abbass, H. A., Ruhul, S., & Newton, C. (2001). PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems. In *Proceedings of the Congress on Evolutionary Computation (CEC'2001)*, IEEE Service Center, Piscataway, NJ (pp. 971-978).
- Abbeel, P. (2008). *Apprenticeship learning and reinforcement learning with application to robotic control*. Unpublished doctoral dissertation, Stanford University.
- Abdelazim, H., & Wahba, K. (2006). An artificial intelligence approach to portfolio selection and management. *International Journal of Financial Services Management*, 1(2-3), 243–254.
- Abdi, H. (2003). Neural networks. In *Encyclopedia of social science research methods*. Thousand Oaks, CA: Sage.
- Abdullah, M. Z., Saleh, J. M., Fathinul-Syahir, A. S., & Mohd-Azemi, B. M. N. (2006). Discrimination and classification of fresh-cut starfruits (*Averrhoa carambola L.*) using automated machine vision system. *Journal of Food Engineering*, 76, 506–523. doi:10.1016/j.jfoodeng.2005.05.053
- Abonyi, J., & Feil, B. (2007). *Cluster analysis for data mining and system identification*. Berlin, Germany: Birkhäuser Basel.
- Abrahamsen, P. (1997). *A review of Gaussian random fields and correlation functions* (Tech. Rep. 917). Norwegian Computing Center, Oslo, Norway.
- Agirre, E., & Martinez, D. (2001). *Knowledge sources for word sense disambiguation*. Paper presented at the 4th International Conference on Text, Speech and Dialogue
- Aha, D., Kibler, D., & Albert, M. (1991). Instance-based learning algorithms. *Machine Learning*, 6(1), 37–66.
- Ahern, D. M., Clouse, A., & Turner, R. (2006). *CMMI distilled*. Reading, MA: Addison-Wesley.
- Ahmad, S., & Tresp, V. (1993). Some solutions to the missing feature problem in vision. [San Mateo, CA: Morgan Kaufmann Publishers Inc.]. *Advances in Neural Information Processing Systems*, 5, 393–400.
- Aickelin, U., & Cayzer, S. (2002). The danger theory and its application to artificial immune systems. In J. Timmis & P.J. Bentley (Eds.), *Proceedings of the International Conference on Artificial Immune Systems (ICARIS)* (pp. 141-148). University of Kent at Canterbury.
- Aickelin, U., Greensmith, J., & Twycross, J. (2004). Immune system approaches to intrusion detection - a review. In *Proceedings of the 3rd International Conference on Artificial Immune Systems* (LNCS 3239, pp. 316-329). Berlin, Germany: Springer.
- Aizerman, M., Braverman, E., & Rozonoer, L. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25, 821–837.

- Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19, 714–723.
- Akaike, H. (1981). Likelihood of a model and information criteria . *Journal of Econometrics*, 16, 3–14.
- Akoka, J., Leune, B., & Koster, A. (1994). Expert system for feasibility assessment of product development. *Expert Systems with Applications*, 7(2), 291–303. doi:10.1016/0957-4174(94)90045-0
- Alba, E., & Tomassini, M. (2002). Parallelism and evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 6(5), 443–462. doi:10.1109/TEVC.2002.800880
- Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., & Walter, P. (2002). *Molecular biology of the cell* (4th ed.). London: Garland Science.
- Alcalá-Fdez, J., Sánchez, L., García, S., Del Jesus, M. J., Ventura, S., & Garrell, J. M. (2009). KEEL: A software tool to assess evolutionary algorithms for data mining problems. *Soft Computing*, 13(3), 307–318. doi:10.1007/s00500-008-0323-y
- Aleixos, N., Blasco, J., Navarrón, F., & Moltó, E. (2002). Multispectral inspection of citrus in real-time using machine vision and digital signal processors. *Computers and Electronics in Agriculture*, 33, 121–137. doi:10.1016/S0168-1699(02)00002-9
- Aler, R., Borrajo, D., & Isasi, P. (2002). Using genetic programming to learn and improve control knowledge. *Artificial Intelligence*, 141, 29–56. doi:10.1016/S0004-3702(02)00246-1
- Allison, P. D. (2001). *Missing data*. Newbury Park, CA: Sage.
- Altman, E. (1968). Financial ratios, discriminant analysis and the prediction of corporate bankruptcy. *The Journal of Finance*, 23, 589–609. doi:10.2307/2978933
- Aluja-Banet, T., & Nonell-Torrent, R. (1991). Local principal component analysis. *Qüestioó*, 3, 267-278.
- Amari, S. I., Cichocki, A., & Yang, H. (1996), A new learning algorithm for blind separation of sources, In Touretzky, Mozer, & Hasselmo (Eds.), *Advances in Neural Information Processing System 8*, MIT Press, 757-763.
- Amir, E. (2006). Learning partially observable action schemas. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence*.
- Amirkian, B., & Nishimura, H. (1994). What size network is good for generalization of a specific task of interest? *Neural Networks*, 7(2), 321–329. doi:10.1016/0893-6080(94)90026-4
- Anagnostopoulos, G., & Georgiopoulos, M. (2001). EllipsoidART and ARTMAP for incremental unsupervised and supervised learning. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'01)*, 2 (pp. 1221-1226).
- Anderberg, M. (1973). *Cluster analysis for applications*. New York: Academic Press.
- Andreevskaia, A., & Bergler, S. (2006). *Mining WordNet for fuzzy sentiment: Sentiment tag extraction from WordNet glosses*. Paper presented at the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL'06).
- Andrews, R., Diederich, J., & Tickle, A. B. (1995). Survey and critique of techniques for extracting rules from trained artificial neural networks. *Knowledge-Based Systems*, 8(6), 373–389. doi:10.1016/0950-7051(96)81920-4
- Anjewierden, A. (2001). AIDAS: Incremental logical structure discovery in pdf documents. In *Proceedings of the 6th International Conference on Document Analysis and Recognition* (pp. 374-378). Washington, DC: IEEE Computer Society.
- Apostolico, A., Comin, M., & Parida, L. (2005). Conservative extraction of over-represented extensible motifs. *Bioinformatics (Oxford, England)*, 21(Suppl 1), i9–i18. doi:10.1093/bioinformatics/bti1051
- Apostolico, A., Gong, F., & Lonardi, S. (2004). Verbumculus and the discovery of unusual words. *Journal*

Compilation of References

- of Computer Science and Technology, 19(1), 22–41. doi:10.1007/BF02944783
- Aramaki, E., & Miyo, K. (2006). Automatic deidentification by using sentence features and label consistency. Paper presented at the i2b2 Workshop on Challenges in Natural Language Processing for Clinical Data.
- Arbib, M. (2002). *The handbook of brain theory and neural networks*. Cambridge, MA: MIT Press.
- Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68, 337–404. doi:10.2307/1990404
- Arthur, D., & Vassilvitskii, S. (2007). K-means++ the advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007*, New Orleans, LA, USA (pp. 1027–1035).
- Asharaf, S., Shevade, S., & Murty, M. (2005). Rough support vector clustering. *Pattern Recognition*, 38, 1779–1783. doi:10.1016/j.patcog.2004.12.016
- Assfalg, J., Bertini, M., Colombo, C., Bimbo, A. d., & Nunziati, W. (2003). Semantic annotation of soccer videos: Automatic highlights identification. *Computer Vision and Image Understanding*, 92(2-3), 285–305. doi:10.1016/j.cviu.2003.06.004
- Atiya, A. F. (2001). Bankruptcy prediction for credit risk using neural networks: A survey and new results. *IEEE Transactions on Neural Networks*, 12, 929–935. doi:10.1109/72.935101
- Atlas, L., Droppo, J., & McLaughlin, J. (1997). Optimizing time-frequency distributions for automatic classification. *SPIE-The International Society for Optical Engineering*, 3162, 161–171.
- Au, K. F., Choi, T. M., & Yu, Y. (2008). Fashion retail forecasting by evolutionary neural networks. *International Journal of Production Economics*, 114, 615–630. doi:10.1016/j.ijpe.2007.06.013
- Auger, F., & Flandrin, P. (1995). Improving the readability of time-frequency and time-scale representations by the reassignment method. *IEEE Transactions on Signal Processing*, 43(5), 1068–1089. doi:10.1109/78.382394
- Auger, F., & Hlawatsch, F. (2008). *Time-frequency analysis*. New York: Wiley-ISTE.
- Aura, T., Kuhn, T., & Roe, M. (2006). Scanning electronic documents for personally identifiable information. Paper presented at the 5th ACM workshop on Privacy in electronic society.
- Ayer, M., Brunk, H., Ewing, G., Reid, W., & Silverman, E. (1955). An empirical distribution function for sampling with incomplete information. *Annals of Mathematical Statistics*, 26(4), 641–647. doi:10.1214/aoms/1177728423
- Bacchus, F., & Ady, M. (2001). Planning with resources and concurrency: A forward chaining approach. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence* (pp. 417–424).
- Bacchus, F., & Kabanza, F. (2000). Using temporal logics to express search control knowledge for planning. *Artificial Intelligence Journal*, 116, 123–191. doi:10.1016/S0004-3702(99)00071-5
- Back, A. (2001). Radial basis functions. In Y. H. Hu & J.-N. Hwang (Eds.), *Handbook of neural network signal processing*. London: CRC Press.
- Back, A. D., & Tsoi, A. C. (1991). FIR and IIR synapses, a new neural network architecture for time series modeling. *Neural Computation*, 3, 375–385. doi:10.1162/neco.1991.3.3.375
- Back, A. D., & Tsoi, A. C. (1992). Nonlinear system identification using multilayer perceptrons with locally recurrent synaptic structure. In *Proceedings of the 1992 IEEE-SP Workshop on Neural Networks for Signal Processing* (Vol. 2, pp. 444–453).
- Baird, H. S. (1994). Background structure in document images. In H. Bunke, P. S. P. Wang, & H. S. Baird (Eds.), *Document image analysis* (pp. 17–34). Singapore: World Scientific.
- Balachandran, S., Dasgupta, D., Nino, F., & Garrett, D. (2007). A framework for evolving multi-shaped detectors

- in negative selection. In *Proceedings of the IEEE Symposium on Foundations of Computational Intelligence*, Honolulu, Hawaii (pp. 401-408).
- Baldi, P., Brunak, S., Chauvin, Y., Andersen, C., & Nielsen, H. (2000). Assessing the accuracy of prediction algorithms for classification: An overview. *Bioinformatics*, 16(5), 412–422. doi:10.1093/bioinformatics/16.5.412
- Banko, M., & Moore, R. (2004). *Part of speech tagging in context*. Paper presented at the 20th International Conference on Computational Linguistics (Coling 2004).
- Barakat, N., & Diederich, J. (2004). Learning-based rule-extraction from support vector machines. In *Proceedings of the 14th International Conference on Computer Theory and applications ICCTA'2004*, Alexandria, Egypt.
- Barakat, N., & Diederich, J. (2005). Eclectic rule-extraction from support vector machines. *International Journal of Computational Intelligence*, 2(1), 59–62.
- Baraldi, A., & Alpaydin, E. (2002). Constructive feed-forward ART clustering networks – part I and II. *IEEE Transactions on Neural Networks*, 13(3), 645–677. doi:10.1109/TNN.2002.1000130
- Baraldi, A., & Blonda, P. (1999). A survey of fuzzy clustering algorithms for pattern recognition – part I and II. *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics*, 29(6), 778–801. doi:10.1109/3477.809032
- Baraniuk, R. G., & Jones, D. L. (1993). A signal-dependent time-frequency representation: Optimal kernel design. *IEEE Transactions on Signal Processing*, 41, 1589–1602. doi:10.1109/78.212733
- Barbounis, T. G., & Theocharis, J. B. (2007a). A locally recurrent fuzzy neural network with application to the wind speed prediction using spatial correlation. *Neurocomputing*, 70(7-9), 1525–1542. doi:10.1016/j.neucom.2006.01.032
- Barbounis, T. G., & Theocharis, J. B. (2007b). Locally recurrent neural networks for wind speed prediction using spatial correlation. *Information Sciences*, 177(24), 5775–5797. doi:10.1016/j.ins.2007.05.024
- Barbounis, T. G., Theocharis, J. B., Alexiadis, M. C., & Dokopoulos, P. S. (2006). Long-term wind speed and power forecasting using local recurrent neural network models. *IEEE Transactions on Energy Conversion*, 21(1), 273–284. doi:10.1109/TEC.2005.847954
- Bartfai, G., & White, R. (1997). ART-based modular networks for incremental learning of hierarchical clusterings. *Connection Science*, 9(1), 87–112. doi:10.1080/095400997116757
- Barzilay, R., & Lapata, M. (2008). Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1), 1–34. doi:10.1162/coli.2008.34.1.1
- Basili, V. R., Green, S., Laitenberger, O., Lanubile, F., Shull, F., Sørumgård, S., & Zelkowitz, M. V. (1996). The empirical investigation of perspective-based reading. *Empirical Software Engineering*, 1(2), 133–164. doi:10.1007/BF00368702
- Bates, J. M., & Granger, C. W. J. (1969). The combination of forecasts. *Operational Research Quarterly*, 20, 451–468. doi:10.1057/jors.1969.103
- Batista, G., & Monard, M. C. (2002). A study of k-nearest neighbour as an imputation method. In *Proceedings of the Soft Computing Systems - Design, Management and Applications, HIS 2002*, Santiago, Chile (pp. 251-260).
- Batista, G., & Monard, M. C. (2003). *Experimental comparison of K-nearest neighbour and mean or mode imputation methods with the internal strategies used by C4.5 and CN2 to treat missing data* (Tech. Rep.). University of Sao Paulo.
- Bau, D., III, & Trefethen, L. N. (1997). *Numerical linear algebra*. Philadelphia, PA: SIAM Press.
- Bauer, F. L. (1957). Das verfahren der treppeniteration und verwandte verfahren zur losung algebraischer eigenwert probleme. *Math. Phys.*, 8, 214–235.
- Baum, E. B., & Haussler, D. (1989). What size net gives valid generalization? *Neural Computation*, 1, 151–160. doi:10.1162/neco.1989.1.1.151
- Baum, L., & Egon, J. (1967). An inequality with applications to statistical estimation for probabilistic functions

Compilation of References

- for a Markov process and to a model for ecology. *Bulletin of the American Meteorological Society*, 73, 360–363. doi:10.1090/S0002-9904-1967-11751-8
- Bazen, A. M., & Veldhuis, R. N. J. (2004). Likelihood-ratio-based biometric verification. *IEEE Trans. Circuits and Systems for Video Technology*, 14(1), 86–94. doi:10.1109/TCSVT.2003.818356
- Beal, M. (2003). *Variational algorithms for approximate Bayesian inference*. Unpublished doctoral dissertation, The Gatsby Computational Neuroscience Unit, University College London.
- Becerra, V. M., Galvao, H., & Abou-Seads, M. (2005). Neural and wavelet network models for financial distress classification. *Data Mining and Knowledge Discovery*, 11, 35–55. doi:10.1007/s10618-005-1360-0
- Becker, J. M. (1985). *Inductive learning of decision rules with exceptions: Methodology and experimentation*. Unpublished bachelor's dissertation, University of Illinois at Urbana-Champaign.
- Belaïd, A., & Rangoni, Y. (2008). Structure extraction in printed documents using neural approaches. In S. Marinai & H. Fujisawa (Eds.), *Machine learning in document analysis and recognition* (pp. 1-24). Berlin, Germany: Springer.
- Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19, 711–720. doi:10.1109/34.598228
- Belkin, M., & Niyogi, P. (2006). Laplacian Eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6), 1373–1396. doi:10.1162/089976603321780317
- Bell, A., & Sejnowski, T. (1995). An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7, 1129–1159.
- Bellman, R. (1957). *Dynamic programming*. Princeton, NJ: Princeton University Press.
- Bellman, R. (1961). *Adaptive control processes: A guided tour*. Princeton, NJ: Princeton University Press.
- Benamara, F., Cesarano, C., Picariello, A., Reforgiato, D., & Subrahmanian, V. (2007). *Sentiment analysis: Adjectives and adverbs are better than the adjectives alone*. Paper presented at the International Conference on Weblogs and Social Media (ICWSM'2007).
- Bender, E. A. (1996). *Mathematical methods in artificial intelligence*. Los Alamitos, CA: IEEE.
- Benediktsson, J. A., Sveinsson, J. R., Ersoy, O. K., & Swain, P. H. (1997). Parallel consensual neural networks. *IEEE Transactions on Neural Networks*, 8, 54–64. doi:10.1109/72.554191
- Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 1(1).
- Bengio, Y., de Mori, R., & Gori, M. (1992). Learning the dynamic of speech with back-propagation for sequences. *Pattern Recognition Letters*, 13, 375–385. doi:10.1016/0167-8655(92)90035-X
- Bengio, Y., Ducharme, R., & Vincent, P. (2001). *A neural probabilistic language model*. Paper presented at the 13th conference on Neural Information Processing Systems (NIPS 2000).
- Ben-Hur, A., Horn, D., Siegelmann, H., & Vapnik, V. (2001). Support vector clustering. *Journal of Machine Learning Research*, 2, 125–137. doi:10.1162/15324430260185565
- Benson, S. (1997). *Learning action models for reactive autonomous agents*. Unpublished doctoral dissertation, Stanford University.
- Bergmann, R., & Wilke, W. (1996). PARIS: Flexible plan adaptation by abstraction and refinement. In *Proceedings of the Workshop on Adaptation in Case-Based Reasoning, ECAI06*.
- Bersini, H. (2002). The immune and chemical crossovers. *IEEE Transactions on Evolutionary Computation*, 6(3), 306–313. doi:10.1109/TEVC.2002.1011543
- Berthold, M. R., & Huber, K. P. (1998). Missing values and learning of fuzzy rules. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(2), 171–178. doi:10.1142/S021848859800015X

- Best, C., van der Goot, E., Blackler, K., Garcia, T., Horby, D., Steinberger, R., & Pouliquen, B. (2005). Mapping world events. In P. Van Oosterom, S. Zlatanova, & E. Fendel (Eds.), *Geo-information for disaster management* (pp. 683-696). Berlin, Germany: Springer.
- Bezdek, J. (1981). *Pattern recognition with fuzzy objective function algorithms*. New York: Plenum Press.
- Bhattacharyya, S., Pictet, O. V., & Zumbach, G. (2002). Knowledge-intensive genetic discovery in foreign exchange markets. *IEEE Transactions on Evolutionary Computation*, 6(2), 169–181. doi:10.1109/4235.996016
- Bi, J., & Bennet, P. (2003). Regression error characteristic curves. In *Proceedings of the 20th International Conference on Machine Learning* (pp. 43-50).
- Biber, D. (1988). Variation across speech and writing. In D. Biber (Ed.), *Variation across speech and writing* (pp. 3-27). Cambridge, UK: Cambridge University Press.
- Bicego, M., Castellani, U., & Murino, V. (2003). Using hidden Markov models and wavelets for face recognition. In *Proc. of the Int. Conf. Image Analysis and Processing* (p. 52).
- Biffl, S., & Grossmann, W. (2001). Evaluating the accuracy of defect estimation models based on inspection data from two inspection cycles. In *Proceedings of the 23rd International Conference on Software Engineering* (pp. 145-154).
- Billard, A., & Schaal, S. (2001). A connectionist model for online robot learning by imitation. In *Proceedings of the 2001 IEEE International Conference on Intelligent Robots and Systems*.
- Bilmes, J. (1997). *A gentle tutorial on the EM algorithm and its application to Parameter estimation for Gaussian mixture and hidden Markov models* (ICSI-Report-97-021).
- Binsztok, H., & Artières, T. (2004). *Learning HMM structure for on-line handwriting modelization*. Paper presented at the IEEE International Workshop on Frontiers in Handwriting Recognition.
- Bishop, C. M. (1995). *Neural networks for pattern recognition*. Gloucestershire, UK: Clarendon Press.
- Bishop, C. M., Svensén, M., & Williams, C. K. I. (1998). GTM: The generative topographic mapping. *Neural Computation*, 10(1), 215–234. doi:10.1162/089976698300017953
- Bishop, C., Hinton, G., & Strachan, I. (1997). GTM through time. In *Proceedings of the IEEE Fifth International Conference on Artificial Neural Networks* (pp.111-116).
- Bishop, C., Svensén, M., & Williams, C. (1998). Developments of the generative topographic mapping. *Neurocomputing*, 21(1-3), 203–224. doi:10.1016/S0925-2312(98)00043-5
- Bishop, J. B., Szalapski, M., Ananthraman, S., McIntrye, D. R., & Pope, M. H. (1997). Classification of low back pain from dynamic motion characteristics using an artificial neural network. *Spine*, 22(24), 2991–2998. doi:10.1097/00007632-199712150-00024
- Bisson, G. (1992a). Learning in FOL with a similarity measure. In W. R. Swartout (Ed.), *Proceedings of the 10th National Conference on Artificial Intelligence – AAAI-92* (pp. 82-87).
- Bisson, G. (1992b). Conceptual clustering in a first order logic representation. In *Proceedings of the 10th European conference on Artificial intelligence*. New York: John Wiley & Sons.
- Blasco, J., Aleixos, N., & Moltó, E. (2003). Machine vision system for automatic quality grading of fruit. *Biosystems Engineering*, 85(4), 415–423. doi:10.1016/S1537-5110(03)00088-6
- Blasco, J., Aleixos, N., & Moltó, E. (2007). Computer vision detection of peel defects in citrus by means of a region oriented segmentation algorithm. *Journal of Food Engineering*, 81, 535–543. doi:10.1016/j.jfoodeng.2006.12.007
- Blasco, J., Aleixos, N., Gómez-Sanchis, J., & Moltó, E. (2007). Citrus sorting identification of the most common defects using multispectral computer vision. *Journal*

Compilation of References

- of Food Engineering, 83, 384–393. doi:10.1016/j.jfoodeng.2007.03.027
- Blasco, J., Aleixos, N., Roger, J. M., Rabatel, G., & Moltó, E. (2002). Robotic weed control using machine vision. *Biosystems Engineering*, 83(2), 149–157. doi:10.1006/bioe.2002.0109
- Blasco, J., Cubero, S., Gómez-Sanchis, J., & Moltó, E. (2008b). Citrus sorting identification of the most common defects using multispectral computer vision. *Lecture Notes in Computer Science*, 5112, 1071–1080. doi:10.1007/978-3-540-69812-8_107
- Blasco, J., Cubero, S., Gómez-Sanchís, J., Mira, P., & Moltó, E. (2008c). Development of a machine for the automatic sorting of pomegranate (*Punica granatum*) arils based on computer vision. *Journal of Food Engineering*, 90(1), 27–34. doi:10.1016/j.jfoodeng.2008.05.035
- Blasco, J., Gómez-Sanchis, J., Gutierrez, A., Chueca, P., Argiles, R., & Moltó, E. (2008a). Automatic sex detection of individuals of Ceratitis Capitata by means of computer vision in a biofactory. *Pest Management Science*, 65(1), 99–104. doi:10.1002/ps.1652
- Blatak, J., Mrakova, E., & Popelinsky, L. (2004). *Fragments and text classification*. Paper presented at the Association for Computational Linguistics (ACL-2004).
- Bloch, G., Sirou, F., Eustache, V., & Fatrez, P. (1997). Neural intelligent control for a steel plant. *IEEE Transactions on Neural Networks*, 8(4), 910–918. doi:10.1109/72.595889
- Blockeel, H., & De Raedt, L. (1998). Top-down induction of first-order logical decision trees. *Artificial Intelligence*, 101, 285–297. doi:10.1016/S0004-3702(98)00034-4
- Blockeel, H., Raedt, L. D., & Ramon, J. (1998). Top-down induction of clustering trees. In J. Shavlik (Ed.), *Proceedings of the 15th International Conference on Machine Learning*. San Francisco: Morgan Kaufmann.
- Blum, A. L., & Furst, M. L. (1995). Fast planning through planning graph analysis. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada (pp. 1636–1642).
- Bobicev, V., & Sokolova, M. (2008). *An effective and robust method for short text classification*. Paper presented at the Twenty-Third Conference on Artificial Intelligence (AAAI 2008).
- Boehm, B. W. (1981). *Software engineering economics*. Englewood Cliffs, NJ: Prentice-Hall.
- Boiy, E., Hens, P., Deschacht, K., & Moens, M.-F. (2007). *Automatic sentiment analysis in on-line text*. Paper presented at the Conference on Electronic Publishing (ELPUB 2007).
- Bolle, R. M., Connell, J. H., Pankanti, S., Ratha, N. K., & Senior, A. W. (2004). *Guide to biometrics*. New York: Springer-Verlag.
- Bonet, B., & Geffner, H. (2001). Planning as heuristic search. *Artificial Intelligence*, 129, 5–33. doi:10.1016/S0004-3702(01)00108-4
- Boquete, L., Barea, R., García, R., Mazo, M., & Espinosa, F. (1999a). Identification and control of a wheelchair using recurrent neural networks. *Engineering Applications of Artificial Intelligence*, 12(4), 443–452. doi:10.1016/S0952-1976(99)00021-4
- Boquete, L., Barea, R., García, R., Mazo, M., & Sotelo, M. A. (2005). Control of a robotic wheelchair using recurrent networks. *Autonomous Robots*, 18, 5–20. doi:10.1023/B:AURO.0000047285.40228.eb
- Boquete, L., García, R., Barea, R., & Mazo, M. (1999). Neural control of the movements of a wheelchair. *Journal of Intelligent & Robotic Systems*, 25(3), 213–226. doi:10.1023/A:1008068322312
- Borrajo, D., & Veloso, M. (1997). Lazy incremental learning of control knowledge for efficiently obtaining quality plans. *AI Review Journal*, 11, 371–340.
- Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In D. Haussler (Ed.), *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory* (pp. 144–152).
- Botea, A., Enzenberger, M., Muller, M., & Schaeffer, J. (2004). Macro-FF: Improving AI planning with auto-

- matically learned macro-operators. *Journal of Artificial Intelligence Research*, 24, 581–621.
- Botros, S. M., & Atkeson, C. G. (1991). Generalization properties of radial basis function, In Lippmann, Moody, & Touretzky (eds), Advances in Neural Information Processing System 3, Morgan Kaufmann Pub., 707-713.
- Box, G. E. P., & Jenkins, G. M. (1970). *Time series analysis -forecasting and control*. San Francisco: Holden Day.
- Bozdogan, H. (1987). Model Selection and Akaike's Information Criterion: The general theory and its analytical extension . *Psychometrika*, 52, 345–370.
- Branco, P. J. C., Dente, J. A., & Mendes, R. V. (2003). Using immunology principles for fault detection. *IEEE Transactions on Industrial Electronics*, 50(2), 362–373. doi:10.1109/TIE.2003.809418
- Brand, M. (1999). Structure learning in conditional probability models via an entropic prior and parameter extinction. *Neural Computation*, 11(5), 1155–1182. doi:10.1162/089976699300016395
- Brand, M., Oliver, N., & Pentland, A. (1997). *Coupled hidden Markov models for complex action recognition*. Paper presented at the IEEE Int'l Conference on Computer Vision and Pattern Recognition.
- Braverman, E. M. (1970). Methods of extremal grouping of parameters and problem of apportionment of essential factors. *Automation and Remote Control*, (1), 108-116.
- Brazma, A., Jonassen, I., Eidhammer, I., & Gilbert, D. (1998). Approaches to the automatic discovery of patterns in biosequences. *Journal of Computational Biology*, 5(2), 279–305. doi:10.1089/cmb.1998.5.279
- Brazma, A., Jonassen, I., Vilo, J., & Ukkonen, E. (1998). Pattern discovery in biosequences. In *Proceedings of the 4th International Colloquium on Grammatical Inference* (LNAI 1433, pp. 255-270). Berlin, Germany: Springer.
- Brazma, A., Parkinson, H., Schlitt, T., & Shojatalab, M. (2001). A quick introduction to elements of biology - cells, molecules, genes, functional genomics microarrays. Retrieved October 2001, from <http://www.ebi.ac.uk/2can/tutorials/index.html>
- Breck, E., Choi, Y., & Cardie, C. (2007). *Identifying expressions of opinion in context*. Paper presented at the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-2007).
- Breiman, L. (1996). Bagging predictors. *Machine Learning*, 24, 123–140.
- Bresina, J., Jonsson, A., Morris, P., & Rajan, K. (2005). Activity planning for the Mars exploration rovers. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- Breuel, T. M. (2002). Two geometric algorithms for layout analysis. In *Proceedings of the 5th Workshop on Document Analysis Systems*.
- Briand, L. C., El-Emam, K., & Freimut, B. G. (1998). A comparison and integration of capture-recapture models and the detection profile method. In *Proceedings of the 9th International Symposium on Software Reliability Engineering* (pp. 32-41).
- Briand, L. C., El-Emam, K., Freimut, B. G., & Laitenberger, O. (2000). A comprehensive evaluation of capture-recapture models for estimating software defect content. *IEEE Transactions on Software Engineering*, 26(6), 518–540. doi:10.1109/32.852741
- Brier, M. E., Zurada, J. M., & Aronoff, G. R. (1995). Neural network predicted peak and trough gentamicin concentrations. *Pharmaceutical Research*, 12(3), 406–412. doi:10.1023/A:1016260720218
- Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. *Computational Linguistics*, 21(4), 543–565.
- Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging . *Computational Linguistics*, 21(4), 543–565.
- Broomhead, D. S., & Lowe, D. (1988). Multivariable functional interpolation and adaptive networks . *Complex Systems*, 2, 321–323.

Compilation of References

- Brox Røst, T., Nytrø, Ø., & Grimsmo, A. (2006). *Classifying encounter notes in the primary care patient record*. Paper presented at the ECAI'06 3rd International Workshop on Text-based Information Retrieval.
- Buhler, J., & Tompa, M. (2001). Finding Motifs using random projections. In *Proceedings of the 5th International Conference on Computational Molecular Biology* (pp. 69-76).
- Bullen, R. J., Cornford, D., & Nabney, I. T. (2003). Outlier detection in scatterometer data: Neural network approaches. *Neural Networks*, 16(3-4), 419–426. doi:10.1016/S0893-6080(03)00013-3
- Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2), 121–167. doi:10.1023/A:1009715923555
- Burget, R. (2007). Layout based information extraction from HTML documents. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 624-628). Washington, DC: IEEE Computer Society.
- Burks, T. F., Shearer, S. A., & Payne, F. A. (2000a). Classification of weed species using color texture features and discriminant analysis. *Transactions of the American Society of Agricultural Engineers*, 43(2), 441–448.
- Bylander, T. (1991). Complexity results for planning. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, Sydney, Australia (pp. 274-279).
- Cadini, F., Zio, E., & Pedroni, N. (2007). Simulating the dynamics of the neutron flux in a nuclear reactor by locally recurrent neural networks. *Annals of Nuclear Energy*, 34, 483–495. doi:10.1016/j.anucene.2007.02.013
- Cadini, F., Zio, E., & Pedroni, N. (2008). Validation of infinite impulse response multilayer perceptron for modelling nuclear dynamics. *Science and Technology of Nuclear Installations*, 2008, 1–10. doi:10.1155/2008/681890
- Calic, J., Campbell, N., Dasiopoulou, S., & Kompatsiaris, Y. (2005, December). *An overview of multimodal video representation for semantic analysis*. Paper presented at the IEEE European Workshop on the Integration of Knowledge, Semantics and Digital Media Technologies.
- Califano, A. (2000). SPLASH: Structural pattern localization analysis by sequential histograms. *Bioinformatics (Oxford, England)*, 16(4), 341–357. doi:10.1093/bioinformatics/16.4.341
- Campbell, M. (Ed.). (2006). *IBM research TRECVID-2006 video retrieval system*.
- Campolucci, P., Uncini, A., & Piazza, F. (1996, May). Fast adaptive IIR-MLP neural networks for signal processing application. In *Proceedings of the IEEE International Conference on Acoustic Speech and Signal Processing, ICASSP'96*, Atlanta, USA (pp. 3530-3533).
- Campolucci, P., Uncini, A., Piazza, F., & Rao, B. D. (1999). On-line learning algorithms for locally recurrent neural networks. *IEEE Transactions on Neural Networks*, 10(2), 253–271. doi:10.1109/72.750549
- Canbas, S. C., & Kilic, S. B. (2005). Prediction of commercial bank failure via multivariate statistical analysis of financial structures: The Turkish case. *European Journal of Operational Research*, 166, 528–546. doi:10.1016/j.ejor.2004.03.023
- Cannas, B., Celli, G., Marchesi, M., & Pilo, F. (1998). Neural networks for power system condition monitoring and protection. *Neurocomputing*, 23, 111–123. doi:10.1016/S0925-2312(98)00065-4
- Cannas, B., Cincotti, S., Fanni, A., Marchesi, M., Pilo, F., & Usai, M. (1998). Performance analysis of locally recurrent neural networks. *COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, 17(6), 708–716. doi:10.1108/03321649810221251
- Cao, G., Gao, J., Nie, J.-Y., & Bai, J. (2007). *Extending query translation to cross-language query expansion with Markov chain models*. Paper presented at the Conference on Information and Knowledge Management (CIKM 2007).
- Carmagnac, F., Héroux, P., & Trupin, E. (2004). Multi-view hac for semi-supervised document image classification.

- tion. In S. Marinai & A. Dengel (Eds.), *Proceedings of the 6th International Workshop on Document Analysis Systems* (pp. 191-200). Berlin, Germany: Springer.
- Carmagnac, F., Héroux, P., & Trupin, E. (2004). Distance based strategy for document image classification. In *Advances in pattern recognition* (pp. 894-902). Berlin, Germany: Springer.
- Carney, M., & Cunningham, P. (2006). Making good probability estimates for regression. In *Proceedings of the 17th European Conference on Machine Learning* (LNCS 4212, pp. 582-589). Berlin, Germany: Springer.
- Carney, M., Cunningham, P., & Lucey, B. M. (2006). Making density forecasting models statistically consistent. *IIS Discussion Paper Series*.
- Carpenter, G., & Grossberg, S. (1987). A massively parallel architecture for a self-organizing neural pattern recognition machine. *Computer Vision Graphics and Image Processing*, 37, 54–115. doi:10.1016/S0734-189X(87)80014-2
- Carpenter, G., & Grossberg, S. (1987). ART2: Self-organization of stable category recognition codes for analog input patterns. *Applied Optics*, 26(23), 4919–4930. doi:10.1364/AO.26.004919
- Carpenter, G., & Grossberg, S. (1988). The ART of adaptive pattern recognition by a self-organizing neural network. *IEEE Computer*, 21(3), 77–88.
- Carpenter, G., & Grossberg, S. (1990). ART3: Hierarchical search using chemical transmitters in self-organizing pattern recognition architectures. *Neural Networks*, 3(23), 129–152. doi:10.1016/0893-6080(90)90085-Y
- Carpenter, G., Grossberg, S., & Rosen, D. (1991). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4, 759–771. doi:10.1016/0893-6080(91)90056-B
- Carreira-Perpiñan, M. A. (2000). Reconstruction of sequential data with probabilistic models and continuity constraints. In S. Solla, T. Leen, & K. R. Müller (Eds.), *Advances in neural information processing systems 12* (pp. 414-420). San Francisco, CA: Morgan Kauffmann.
- Carrick, C., Yang, Q., Abi-zeid, I., & Lamontagne, L. (1999). Activating CBR systems through autonomous information gathering. In *Proceedings of the 3rd International Conference on Case-Based Reasoning*.
- Carse, B., Fogarty, T. C., & Munro, A. (1996). Evolving fuzzy rule based controllers using genetic algorithms. *Fuzzy Sets and Systems*, 80, 273–293. doi:10.1016/0165-0114(95)00196-4
- CART. (2009). *Salford Systems Inc*. Retrieved from <http://www.salford-systems.com>
- Caruana, R. (1997). *Multitask learning*. Unpublished doctoral dissertation, Carnegie Mellon University.
- Caruana, R., & Niculescu-Mizil, A. (2004). Data mining in metric space: An empirical analysis of supervised learning performance criteria. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 69-78).
- Carvalho, V., & Cohen, W. (2004). *Learning to extract signature and reply lines from email*. Paper presented at the First Conference on Email and Anti-Spam.
- Castel, J. M., Mena, Y., Delgado-Pertiñez, M., Carmuñez, J., Basalto, J., & Caravaca, F. (2003). Characterization of semi-extensive goat production systems in southern. *Small Ruminant Research*, 47, 133–143. doi:10.1016/S0921-4488(02)00250-X
- Castillo, L., Fdez-Olivares, J., García-Pérez, O., & Palao, F. (2006). Bringing users and planning technology together. Experiences in SIADEX. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling*.
- Caudell, T., Smith, S., Escobedo, R., & Anderson, M. (1994). NIRS: Large scale ART-1 neural architectures for engineering design retrieval. *Neural Networks*, 7(9), 1339–1350. doi:10.1016/0893-6080(94)90084-1
- Caudell, T., Smith, S., Johnson, G., & Wunsch, D., II. (1991). An application of neural networks to group technology. In *Proceedings of SPIE, vol. 1469, Applications of Neural Networks II* (pp. 612-621).

Compilation of References

- Cavalieri, S., Maccarrone, P., & Pinto, R. (2004). Parametric vs. neural network models for the estimation of production costs: A case study in the automotive industry. *International Journal of Production Economics*, 91(2), 165–177. doi:10.1016/j.ijpe.2003.08.005
- Cavanaugh, J. (1997). Unifying the derivations for the Akaike and corrected Akaike information criteria . *Statistics & Probability Letters*, 33, 201–208.
- Ceri, S., Gottlob, G., & Tanca, L. (1990). *Logic programming and databases*. Berlin, Germany: Springer.
- Cestnik, B. (1990). Estimating probabilities: A crucial task in machine learning. In *Proceedings of the 9th European Conference on Artificial Intelligence* (pp. 147-149).
- Chan, L. W., & Young, E. F.-Y. (1995, June). *Locally connected recurrent networks* (Tech. Rep. CS-TR-95-10). Hong Kong: The Chinese University of Hong Kong New Territories, Computer Science Department.
- Chan, Y., Ng, H., & Chiang, D. (2007). *Word sense disambiguation improves statistical machine translation*. Paper presented at the Association for Computational Linguistics (ACL-2007).
- Chang, P. R., & Yang, W. H. (1997). Environment-adaptation mobile radio propagation prediction using radial basis function neural networks . *IEEE Transactions on Vehicular Technology*, 46, 155–160.
- Chang, P., Han, M., & Gong, Y. (2002). *Highlight detection and classification of baseball game video with hidden Markov models*. Paper presented at the IEEE Int'l Conference on Image Processing, Rochester, NY.
- Chang, S. F., & Hsu, W. (Eds.). (2006). *Columbia University TRECVID-2006 video search and high-level feature extraction*.
- Chao, H. (2003). Graphics extraction in PDF document. In T. Kanungo, E. H. B. Smith, J. Hu, & P. B. Kantor (Eds.), *SPIE - The International Society for Optical Engineering. Volume 5010*, (pp. 317-325).
- Chao, H., & Fan, J. (2004). Layout and content extraction for pdf documents. In S. Marinai & A. Dengel (Eds.), *Proceedings of the 6th International Workshop on Document Analysis Systems* (pp. 213-224). Berlin: Springer.
- Chao, H., & Lin, X. (2005). Capturing the layout of electronic documents for reuse in variable data printing. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 940-944). Washington, DC: IEEE Computer Society.
- Chapelle, O., Vapnik, V., Bousquet, O., & Mukherjee, S. (2002). Choosing multiple parameters for support vector machines. *Machine Learning*, 46(1/3), 131–159. doi:10.1023/A:1012450327387
- Chapman, P., Clinton, J., Kerber, R., Khabaza, T., Reinartz, T., Shearer, C., & Wirth, R. (2000). *CRISP-DM 1.0: Step-by-step data mining guide*. Retrieved from <http://www.crisp-dm.org>
- Chappell, G., & Taylor, J. (1993). The temporal Kohonen map. *Neural Networks*, 6, 441–445. doi:10.1016/0893-6080(93)90011-K
- Charniak, E. (2000). *A maximum entropy inspired parser*. Paper presented at the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2000).
- Charniak, E. (2000). *A maximum entropy inspired parser*. Paper presented at the First Annual Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL 2000).
- Chatfield, C. (1993). Neural networks: Forecasting breakthrough or passing fad? *International Journal of Forecasting*, 9, 1–3. doi:10.1016/0169-2070(93)90043-M
- Chatzis, V., Bors, A. G., & Pitas, I. (1999). Multimodal decision-level fusion for person authentication. *IEEE Trans. System, Man, and Cybernetics, part A*, 29(6), 674–680.
- Chauvin, Y., & Rumelhart, D. (1995). *Backpropagation: Theory, architectures, and applications*. Hillsdale, NJ: Lawrence Erlbaum.
- Chaves, A. C. F., Vellasco, M. M. B. R., & Tanscheit, R. (2005). *Fuzzy rule extraction from support vector machines*. Paper presented at the Fifth International Conference on Hybrid Intelligent Systems.

- Chen, C.-L., Chen, W.-C., & Chang, F.-Y. (1993). Hybrid learning algorithm for Gaussian potential function networks. *IEEE Proceedings. Part D. Control Theory and Applications*, 140(6), 442–448.
- Chen, L. F., Liao, H. Y. M., & Lin, J. C. (2001). Person identification using facial motion. In *Proc. of the Int. Conf. Image Processing* (pp. 677-680).
- Chen, S., & Lu, C. (1999). Would evolutionary computation help in designs of ANNs in forecasting foreign exchange rates? In *Proceedings of the 1999 Congress on Evolutionary Computation, Vol. 1* (pp. 267-274).
- Chen, S., Cowan, C. N., & Grant, P. M. (1991). Orthogonal least squares learning algorithm for Radial basis function networks . *IEEE Transactions on Neural Networks*, 2, 302–309.
- Chen, S., Mao, S., & Thoma, G. (2007). Simultaneous layout style and logical entity recognition in a heterogeneous collection of documents. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 118-122). Washington, DC: IEEE Computer Society.
- Chen, W. P., Tang, F. T., & Ju, C. W. (2001). Stress distribution of the foot during mid-stance to push-off in barefoot gait: A 3-D finite element analysis. *Clinical Biomechanics (Bristol, Avon)*, 16, 614–620. doi:10.1016/S0268-0033(01)00047-X
- Cheung, J. T., Zhang, M., Leung, A. K., & Fan, Y. B. (2005). Three-dimensional finite element analysis of the foot during standing-a material sensitivity study. *Journal of Biomechanics*, 38, 1045–1054. doi:10.1016/j.jbiomech.2004.05.035
- Chiang, J., & Hao, P. (2003). A new kernel-based fuzzy clustering approach: Support vector clustering with cell growing. *IEEE transactions on Fuzzy Systems*, 11(4), 518–527. doi:10.1109/TFUZZ.2003.814839
- Cho, J. S., Kim, Y. W., & Park, D. J. (1997). Identification of nonlinear dynamic systems using higher order diagonal recurrent neural network. *IEEE Electronics Letters*, 33(25), 2133–2135. doi:10.1049/el:19971398
- Choi, T. M. (2007). Pre-season stocking and pricing decisions for fashion retailers with multiple information updating. *International Journal of Production Economics*, 106, 146–170. doi:10.1016/j.ijpe.2006.05.009
- Choi, Y., Cardie, C., Riloff, E., & Patwardhan, S. (2005). *Identifying sources of opinions with conditional random fields and extraction patterns*. Paper presented at the Empirical Methods for Natural Language Processing (EMNLP 2005).
- Choudhary, R., Paliwal, J., & Jayas, D. S. (2008). Classification of cereal grains using wavelet, morphological, colour, and textural features of non-touching kernel images. *Biosystems Engineering*, 99, 330–337. doi:10.1016/j.biosystemseng.2007.11.013
- Chtioui, Y., Panigrahi, S., & Backer, L. F. (1999). Rough sets theory as a pattern classification tool for quality assessment of edible beans. *Transactions of the American Society of Agricultural Engineers*, 42(4), 1145–1152.
- Chu, C. W., & Zhang, G. P. (2003). A comparative study of linear and nonlinear models for aggregate retail sales forecasting. *International Journal of Production Economics*, 86(3), 217–231. doi:10.1016/S0925-5273(03)00068-9
- Chua, L. O. (1993). Global unfolding of Chua's circuit. *IEICE Transactions on Fundamentals E (Norwalk, Conn.)*, 76-A, 704–734.
- Chua, L. O., & Kang, S. M. (1976). Memristive devices and systems. *Proceedings of the IEEE*, 64, 209–223. doi:10.1109/PROC.1976.10092
- Chua, L. O., & Lin, G. N. (1990). Canonical realisation of Chua's circuit family. *IEEE Transactions on Circuits and Systems*, 37(7), 885–902. doi:10.1109/31.55064
- Chua, L. O., Komuro, M., & Matsumoto, T. (1986). The double scroll family. *IEEE Transactions on Circuits and Systems*, 33(11), 1072–1118. doi:10.1109/TCS.1986.1085869
- Chung, K.-M., Kao, W.-C., Sun, C.-L., Wang, L.-L., & Lin, C.-J. (2003). Radius margin bounds for support vector machines with the RBF kernel. *Neural Computation*, 15(11), 2643–2681. doi:10.1162/089976603322385108

Compilation of References

- Ciocoiu, I. B. (1996). RBF networks with FIR/IIR synapses. *Neural Processing Letters*, 3(1), 17–22. doi:10.1007/BF00417785
- Ciocoiu, I. B. (1998). Time series analysis using RBF networks with FIR/IIR synapses. *Neurocomputing*, 20, 57–66. doi:10.1016/S0925-2312(98)00024-1
- Clark, P., & Niblett, T. (1982). The CN2 induction algorithm. *Machine Learning*, 3(4), 261–283.
- Clemen, R. (1989). Combining forecasts: A review & annotated bibliography with discussion. *International Journal of Forecasting*, 5, 559–608. doi:10.1016/0169-2070(89)90012-5
- Clerc, M. (2005). *Particle swarm optimization*. Washington, DC: ISTE Press.
- Cliff, D., Harvey, I., & Husbands, P. (1993). Explorations in evolutionary robotics. *Adaptive Behavior*, 2(1), 73–110. doi:10.1177/105971239300200104
- Climer, S., & Zhang, W. (2006). Rearrange clustering: Pitfalls, remedies, and applications. *Journal of Machine Learning Research*, 7, 919–943.
- Coakley, J. R. (1995). Using pattern analysis methods to supplement attention-directing analytical procedures. *Expert Systems with Applications*, 9(4), 513–528. doi:10.1016/0957-4174(95)00021-6
- Cochran, R. N., & Horne, F. H. (1977). Statistically weighted principal component analysis of rapid scanning wavelength kinetics experiments. *Analytical Chemistry*, 49, 846–853. doi:10.1021/ac50014a045
- Coello, C. A. C. (2004). Handling multiple objectives with particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, 8(3), 256–279. doi:10.1109/TEVC.2004.826067
- Coello, C. A. C. (2005). Recent trends in evolutionary multiobjective optimization: Theoretical advances and applications. In *Evolutionary multiobjective optimization* (pp. 7-32). Berlin, Germany: Springer-Verlag.
- Coello, C. A. C., & Cortés, N. C. (2005). Solving multiobjective optimization problems using an artificial immune system. *Genetic Programming and Evolvable Machines*, 6(2), 163–190. doi:10.1007/s10710-005-6164-x
- Coello, C. A. C., Pulido, G. T., & Montes, E. M. (2005). Current and future research trends in evolutionary multiobjective optimization. In *Information processing with evolutionary algorithms* (pp. 213-231). Berlin, Germany: Springer.
- Cohen, L. (1989). Time-frequency distributions-a review. *Proceedings of the IEEE*, 77(7), 941–981. doi:10.1109/5.30749
- Cohen, W. W. (1990). Learning approximate control rules of high utility. In *Proceedings of the International Conference on Machine Learning* (pp. 268-276).
- Colas, F., & Brazdil, P. (2006). Comparison of SVM and some older classification algorithms in text classification tasks. In M. Bramer (Ed.), *Artificial intelligence in theory and practice* (pp. 169-178). Berlin, Germnay: Springer.
- Cole, R., & Gunther, J. (1995). A CAMEL rating's shelf life. *Federal Reserve Bank of Dallas Review, December*, 13-20.
- Coles, A., & Smith, A. (2007). Marvin: A heuristic search planner with online macro-action learning. *Journal of Artificial Intelligence Research*, 28, 119–156.
- Collier, N., Kawazoe, A., & Jin, L. (2007). A multilingual ontology for infectious disease outbreak surveillance: Rationale, design and challenges. *Journal of Language Resources and Evaluation*, 40(3-4), 405–413. doi:10.1007/s10579-007-9019-7
- Collins, M., Hajic, J., Ramshaw, L., & Tillmann, C. (1999). *A statistical parser for Czech*. Paper presented at the Association for Computational Linguistics (ACL-99).
- Collobert, R., & Weston, J. (2008). *A unified architecture for natural language processing: Deep neural networks with multitask learning*. Paper presented at the 25th International Conference on Machine Learning (ICML-2008).
- Cooke, M., Green, P., & Crawford, M. (1994). Handling missing data in speech recognition. In *Proceedings of the*

- International Conference on Spoken Language Processing*, Yokohama, Japan (pp. 1555-1558).
- Cooper, N. (1994). *The Human genome project, deciphering the blueprint of heredity*. New York: University Science Books.
- Cortes, C., & Vapnik, V. (1995). Support vector networks. *Machine Learning*, 20, 273–297.
- Cortez, P., Rocha, M., & Neves, J. (2006). Time series forecasting by evolutionary neural networks. In *Artificial neural networks in real-life applications* (pp. 47-70).
- Cortez, P., Rocha, M., Machado, J., & Neves, J. (1995). A neural network based forecasting system. In *Proceedings of the ICNN'95 – IEEE Int. Conf. on Neural Networks*, Perth, Western Australia (pp. 2689-2693).
- Cottrell, M., Hammer, B., Hasenfu, A., & Villmann, T. (2006). Batch and median neural gas. *Neural Networks*, 19(6-7), 762–771. doi:10.1016/j.neunet.2006.05.018
- Cover, T. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE Transactions on Electronic Computers*, EC-14, 326–334. doi:10.1109/PGEC.1965.264137
- Cover, T. M., & Thomas, J. A. (1991). *Elements of information theory*. New York: Wiley.
- Craven, M. W., & Shavlik, J. W. (1994). Using sampling and queries to extract rules from trained neural networks. In *Machine Learning: Proceedings of the Eleventh International Conference*, San Francisco, CA, USA.
- Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge, UK: Cambridge University Press.
- Cristianini, N., Kandola, J., Elisseeff, A., & Shawe-Taylor, J. (2006). On kernel target alignment. In *Innovations in machine learning* (Vol. 194, pp. 205-256). Berlin, Germany: Springer.
- Crystal, D. (2006). *Language and the Internet*. Cambridge, UK: Cambridge University Press.
- Cucker, F., & Smale, S. (2002). On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39(1), 1–49. doi:10.1090/S0273-0979-01-00923-5
- Curry, B., & Moutinho, L. (1993). Neural networks in marketing: Modelling consumer responses to advertising stimuli. *European Journal of Marketing*, 27(7). doi:10.1108/03090569310040325
- Cutting, D., Karger, D., Pedersen, J., & Turkey, J. (1992). *A cluster-based approach to browsing large document collections*. Paper presented at the Annual International Conference of Special Interest Group on Information Retrieval (SIGIR-92).
- Cyberbotics. (2003). *Distance sensors random noise*. Retrieved from <http://www.cyberbotics.com/cdrom/common/doc/webots/reference/section2.15.html>
- Cyberbotics. (2003). *WEBOTS simulator*. Retrieved from <http://www.cyberbotics.com/>
- Daelemans, W. (2006). *A mission for computational natural language learning*. Paper presented at the 10th Conference on Computational Natural Language Learning (CoNLL-X).
- Daelemans, W., Zavrel, J., van der Sloot, K., & van der Bosch, A. (1999). *TiMBL: Tilburg memory base learner, version 2.0, reference guide* (Tech. Rep.). The Netherlands: University of Tilburg.
- Dagli, C. H., & Sittisathanchai, S. (1995). Genetic neuro-scheduler: A new approach for job shop scheduling. *International Journal of Production Economics*, 41(1-3), 135–145. doi:10.1016/0925-5273(95)00072-0
- Darbari, A. (2001). *Rule extraction from trained ANN: A survey (research index)*. Institute of Artificial Intelligence, Dept. of Computer Science, TU Dresden, Germany.
- Das, M., & Dai, H. (2007). A survey of DNA motif finding algorithms. *BMC Bioinformatics*, 8(Suppl 7), S21. doi:10.1186/1471-2105-8-S7-S21
- Das, S. (2008). Nelder-Mead evolutionary hybrid algorithms. In J. Dopico, J. de la Calle, & A. Sierra (Eds.),

Compilation of References

- Encyclopedia of artificial intelligence* (pp. 1191-1196). Hershey, PA: Information Science Reference.
- Das, S., & Panigrahi, P. K. (2008). Evolutionary algorithms for multi-objective optimization. In J. Dopico, J. de la Calle, & A. Sierra (Eds.), *Encyclopedia of artificial intelligence* (pp. 1145-1151). Hershey, PA: Information Science Reference.
- Das, S., Gui, M., & Pahwa, A. (2008). Artificial immune systems for self-nonself discrimination: Application to anomaly detection. *Advances of Computational Intelligence in Industrial Systems*, 116, 231–248. doi:10.1007/978-3-540-78297-1_11
- Das, S., Koduru, P., Welch, S. M., Gui, M., Cochran, M., Wareing, A., & Babin, B. (2006). Adding local search to particle swarm optimization. In *Proceedings of the World Congress on Computational Intelligence*, Vancouver, BC, Canada (pp. 428-433).
- Das, S., Natarajan, B., Stevens, D., & Koduru, P. (2008). Multi-objective and constrained optimization for DS-CDMA code design based on the clonal selection principle. *Applied Soft Computing Journal*, 8, 788–797. doi:10.1016/j.asoc.2007.05.012
- Dasgupta, D., & Gonzalez, F. (2002). An immunity-based technique to characterize intrusions in computer networks. *IEEE Transactions on Evolutionary Computation*, 6(3), 281–291. doi:10.1109/TEVC.2002.1011541
- Dasgupta, D., & Gonzalez, F. (2003). Anomaly detection using real-valued negative selection. *Journal of Genetic Programming and Evolvable Machines*, 4(4), 383–403. doi:10.1023/A:1026195112518
- Dasgupta, D., Krishna-Kumar, K., Wong, D., & Berry, M. (2004). Negative selection algorithm for aircraft fault detection. In *Artificial immune systems*, (LNCS 3239, pp. 1-13). Berlin, Germany: Springer.
- Dasgupta, D., Yu, S., & Majumdar, N. (2005). MILA -- multi-level immune learning algorithm and its application to anomaly detection. *The Soft Computing Journal*, 9(3), 172–184. doi:10.1007/s00500-003-0342-7
- Dash, P. K., Panigrahi, B. K., & Panda, G. (2003). Power quality analysis using S-transform. *IEEE Transactions on Power Delivery*, 18(2), 406–411. doi:10.1109/TPWRD.2003.809616
- Daugman, J. (1988). Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36, 1169–1179. doi:10.1109/29.1644
- Davis, R., Buchanan, B. G., & Shortliffe, E. (1977). Production rules as a representation for a knowledge-based consultation program. *Artificial Intelligence*, 8(1), 15–45. doi:10.1016/0004-3702(77)90003-0
- Davy, M., & Doncarli, C. (1998). Optimal kernels of time-frequency representations for signal classification. In *Proceedings of the IEEE International Symposium on TIFS* (pp. 581-584).
- Davy, M., Doncarly, C., & Boudreux-Bartels, G. (2001). Improved optimization of time-frequency based signal classifiers. *IEEE Signal Processing Letters*, 8(2), 52–57. doi:10.1109/97.895373
- Davy, M., Gretton, A., Doucet, A., & Rayner, P. W. (2002). Optimised support vector machines for nonstationary signal classification. *IEEE Signal Processing Letters*, 9(12), 442–445. doi:10.1109/LSP.2002.806070
- Dawson, C., & Silklossy, L. (1977). The role of pre-processing in problem solving system. In *Proceedings of the 5th International Joint Conference on Artificial Intelligence* (pp. 465-471).
- Dayan, P., & Balleine, B. W. (2002). Reward, motivation and reinforcement learning. *Neuron*, 36, 285–298. doi:10.1016/S0896-6273(02)00963-7
- Dayan, P., Hinton, G. E., Neal, R. M., & Zemel, R. S. (1995). The Helmholtz machine . *Neural Computation*, 7(5), 889–904.
- De Castro, L. N., & Timmis, J. (2002). *Artificial immune systems: A new computational intelligence approach*. London: Springer-Verlag.
- De Castro, L. N., & Timmis, J. (2003). Artificial immune systems as a new soft computing paradigm. *Soft Computing - A Fusion of Foundations . Methodologies and Applications*, 7(8), 526–544.

- de Castro, L. N., & Von Zuben, F. J. (2002). Learning and optimization using the clonal selection principle. *IEEE Transactions on Evolutionary Computation*, 6(3), 239–251. doi:10.1109/TEVC.2002.1011539
- de la Rosa, T., Garcia-Olaya, A., & Borrajo, D. (2007). Using cases utility for heuristic planning improvement. In *Proceedings of the 7th International Conference on Case-Based Reasoning*, Belfast, Northern Ireland.
- de la Rosa, T., Jimenez, S., & Borrajo, D. (2008). Learning relational decision trees for guiding heuristic planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- deLemos, R., Timmis, J., Ayara, M., & Forrest, S. (2007). Immune-inspired adaptable error detection for automated teller machines. *IEEE Transactions on Systems, Man and Cybernetics. Part C, Applications and Reviews*, 37(5), 873–886. doi:10.1109/TSMCC.2007.900662
- De Toni, A., & Meneghetti, A. (2000). The production planning process for a network of firms in the textile-apparel industry. *International Journal of Production Economics*, 65, 17–32. doi:10.1016/S0925-5273(99)00087-0
- de Vries, B., & Principe, J. (1992). The gamma model – a new neural model for temporal processing. *Neural Networks*, 5(4), 565–576. doi:10.1016/S0893-6080(05)80035-8
- Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley.
- Deb, K. (2007). Evolutionary multi-objective optimization without additional parameters. *Parameter Setting in Evolutionary Algorithm . Studies in Computational Intelligence*, 54, 241–257. doi:10.1007/978-3-540-69432-8_12
- Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2), 182–197. doi:10.1109/4235.996017
- DeGroot, M., & Fienberg, S. (1982). The comparison and evaluation of forecasters. *The Statistician*, 31(1), 12–22.
- Delicado, P. (2001). Another look at principal curves and surfaces. *Journal of Multivariate Analysis*, 77, 84–116. doi:10.1006/jmva.2000.1917
- Dempster, A., Laird, N., & Rubin, D. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B. Methodological*, 39(1), 1–38.
- Dempster, M. A. H., & Leemans, V. (2006). An automated FX trading system using adaptive reinforcement learning. *Expert Systems with Applications*, 30(3), 543–552. doi:10.1016/j.eswa.2005.10.012
- Dengel, A. (2007). Learning of pattern-based rules for document classification. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 123–127). Washington, DC: IEEE Computer Society.
- Devroye, L. (1981). On the almost everywhere convergence of nonparametric regression function estimates . *Annals of Statistics*, 9, 1310–1319.
- Devroye, L. (1987). *A Course in Density Estimation*, Boston: Birkhauser.
- Díaz Blanco, I., Cuadrado Vega, A. A., Diez González, A., Rodríguez Loredo, L., Obeso Carrera, F., & Rodríguez, J. A. (2003). Visual predictive maintenance tool based on SOM projection techniques . *Revue de Métallurgie-Cahiers D Informations Techniques*, 100(3), 307-315.
- Diaz, R., Faus, G., Blasco, M., Blasco, J., & Moltó, E. (2000). The application of fast algorithm for the classification of olives by machine vision. *Food Research International*, 33, 305–309. doi:10.1016/S0963-9969(00)00041-7
- DiCesare, G. (2006). *Imputation, estimation and missing data in finance*. Unpublished doctoral dissertation, University of Waterloo.
- Diday, E. (1979). *Optimisation en classification automatique, Tome 1,2* (in French). Rocquencourt, France: INRIA.
- Diederich, J., & Barakat, N. (2004). Hybrid rule-extraction from support vector machines. In *Proceedings of the*

Compilation of References

- IEEE conference on cybernetics and intelligent systems*, Singapore (pp. 1270-1275).
- Dietterich, T. G., Lathrop, R. H., & Lozano-Perez, T. (1997). Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2), 31–71. doi:10.1016/S0004-3702(96)00034-3
- Ding, Y., & Fan, G. (2006). *Camera view-based American football video analysis*. Paper presented at the Eighth IEEE International Symposium on Multimedia.
- Ding, Y., & Fan, G. (2007). *Segmental hidden Markov models for view-based sport video analysis*. Paper presented at the IEEE Int'l Conference on Computer Vision and Pattern Recognition.
- Ding, Y., & Fan, G. (2007). *Two-layer generative models for sport video mining*. Paper presented at the IEEE International Conference on Multimedia and Expo.
- Ding, Y., & Fan, G. (2008). *Multi-channel segmental hidden Markov models for sports video mining*. Paper presented at the The ACM Multimedia Conference.
- Do, M. B., & Kambhampati, S. (2000). Solving planning graph by compiling it into a CSP. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems (AIPS00)*.
- Doan, S., Hung-Ngo, Q., Kawazoe, A., & Collier, N. (2008). *Global health monitor - a Web-based system for detecting and mapping infectious diseases*. Paper presented at the International Joint Conference on Natural Language Processing (IJCNLP - 2008).
- Domenico, S., & Gary, W. (1994). Machine vision and neural nets in food processing and packaging—natural way combinations. In *Food processing automation III—Proceedings of the FPAC conference* (pp. 11). Orlando, FL: ASAE.
- Domingos, P. (1995). Rule induction and instance-based learning: A unified approach. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence* (pp. 1226-1232). San Francisco, CA: Morgan Kaufmann.
- Domínguez, O., Miranda-Ariz, J. M., & Salvador, L. (2002). Efecto protector de las capas de productos de corrosión de exposición atmosférica. *Revista de Metallurgia*, 38(2), 108–116.
- Dorffner, G. (1996). Neural networks for time series processing. *Neural Networks World*, 6(4), 447–468.
- Dowe, D. L., Farr, G. E., Hurst, A. J., & Lentin, K. L. (1996). Information-theoretic football tipping. In *Proceedings of the 3rd Conference on Maths and Computers in Sport* (pp. 233-241).
- Dredze, M., Blitzer, J., & Pereira, F. (2007). *Biographies, Bollywood, boom-boxes, and blenders: Domain adaptation for sentiment classification*. Paper presented at the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007).
- Driessens, K., & Ramon, J. (2003). Relational instance based regression for relational reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.
- Duan, K., Keerthi, S., & Poo, A. (2003). Evaluation of simple performance measures for tuning svm hyperparameters. *Neurocomputing*, 51, 41–59. doi:10.1016/S0925-2312(02)00601-X
- Duan, L.-Y., Xu, M., Chua, T.-S., Tian, Q., & Xu, C.-S. (2003). *A mid-level representation framework for semantic sports video analysis*. Paper presented at the ACM Multimedia Conference.
- Dubois, D., & Prade, H. (1996). What are fuzzy rules and how to use them. *Fuzzy Sets and Systems*, 84, 169–185. doi:10.1016/0165-0114(96)00066-8
- Duda, R. O., Hart, P. E., & Stork, D. G. (2000). *Pattern classification* (2nd ed.). New York: Wiley-Interscience.
- Dueck, G., & Scheuer, T. (1990). Threshold accepting: A general-purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, 90(1), 161–175. doi:10.1016/0021-9991(90)90201-B
- Dunteman, G. H. (1989). *Principal components analysis*. Newbury Park, CA: Sage Publications.

- Durbin, R., & Willshaw, D. (1987). An analogue approach to the travelling salesman problem using an elastic net method. *Nature*, 326, 689–691. doi:10.1038/326689a0
- Durbin, R., Eddy, S., Krogh, A., & Mitchison, G. (1998). *Biological sequence analysis: Probabilistic models of proteins and nucleic acids*. Cambridge, UK: Cambridge University Press.
- Dzeroski, S., De Raedt, L., & Driessens, K. (2001). Relational reinforcement learning. *Machine Learning*, 43, 7–52. doi:10.1023/A:1007694015589
- Edgar, E. O., Freund, R., & Girosi, F. (1997). *Support vector machines: Training and applications* (Research report). Massachusetts Institute of Technology.
- Edmonds, P. (2002). Introduction to Senseval. *ELRA Newsletter*, 7(3).
- Efron, B. (1967). The two sample problem with censored data. In *Proc. of the Fifth Berkeley Simp. Math. Statist. Probab.* 4 (pp. 831-853). Berkeley, CA: Univ.California Press.
- Egozi, O., Gabrilovich, E., & Markovitch, S. (2008). *Concept-based feature generation and selection for information retrieval*. Paper presented at the Twenty-Third Conference on Artificial Intelligence (AAAI 2008).
- Eick, S. G., Loader, C. R., Long, M. D., Votta, L. G., & Vander Wiel, S. A. (1992). Estimating software fault content before coding. In *Proceedings of the 14th International Conference on Software Engineering* (pp. 59-65).
- Einbeck, J., Evers, L., & Bailer-Jones, C. (2008). Representing complex data using localized principal components with application to astronomical data. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 178-201). Berlin, Germany: Springer.
- Einbeck, J., Tutz, G., & Evers, L. (2005). Local principal curves. *Statistics and Computing*, 15, 301–313. doi:10.1007/s11222-005-4073-8
- Ekin, A., Tekalp, A., & Mehrotra, R. (2003). Automatic soccer video analysis and summarization. *IEEE Transactions on Image Processing*, 12(7), 796–807. doi:10.1109/TIP.2003.812758
- El-Emam, K., & Laitenberger, O. (2001). Evaluating capture-recapture models with two inspectors. *IEEE Transactions on Software Engineering*, 27(9), 851–864. doi:10.1109/32.950319
- Elger, B., & Caplan, A. (2006). Consent and anonymization in research involving biobanks. *European Molecular Biology Organization reports*, 7(7), 661-666.
- Elizondo, D., & Fiesler, E. (1997). A survey of partially connected neural networks. *International Journal of Neural Systems*, 8(5-6), 535–558. doi:10.1142/S0129065797000513
- Emde, W., & Wettschereck, D. (1996). Relational instance based learning. In L. Saitta (Ed.), *Proceedings of the 13th International Conference on Machine Learning* (pp. 122-130).
- Er, M. J. (2002). Face recognition with radial basis function (RBF) neural networks . *IEEE Transactions on Neural Networks*, 13(3), 697–710.
- Erwin, E., Obermayer, K., & Schulten, K. (1992). Self-organizing maps: Ordering, convergence properties and energy functions. *Biological Cybernetics*, 67, 47–55. doi:10.1007/BF00201801
- Eschbach, J. W., & Adamson, J. W. (1985). Anemia of end-stage renal disease (ESRD). *Kidney International*, 28, 1–5. doi:10.1038/ki.1985.109
- Esposito, F., Ferilli, S., Basile, T. M. A., & Di Mauro, N. (2008). Machine learning for digital document processing: From layout analysis to metadata extraction. In S. Marinai & H. Fujisawa (Eds.), *Machine learning in document analysis and recognition* (pp. 79-112). Berlin, Germany: Springer.
- Esposito, F., Malerba, D., & Semeraro, G. (1992). Classification in noisy environments using a distance measure between structural symbolic descriptions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(3), 390–402. doi:10.1109/34.120333

Compilation of References

- Ester, M., & Zhang, X. (2004). A top-down method for mining most specific frequent patterns in biological sequence data. In *Proceedings of the 4th SIAM International Conference on Data Mining*.
- Esuli, A., & Sebastiani, F. (2006). *Determining term subjectivity and term orientation for opinion mining*. Paper presented at the European Chapter of the Association for Computational Linguistics (EACL'07).
- Esuli, A., & Sebastiani, F. (2006). *SENTIWORDNET: A publicly available lexical resource for opinion mining*. Paper presented at the 5th Conference on Language Resources and Evaluation (LREC 2006).
- Esuli, A., & Sebastiani, F. (2007). *Random-walk models of term semantics: An application to opinion-related properties*. Paper presented at the 3rd Language and Technology Conference (LTC 2003).
- Etzioni, O. (1993). Acquiring search-control knowledge via static analysis. *Artificial Intelligence*, 62, 255–301. doi:10.1016/0004-3702(93)90080-U
- Everitt, B., Landau, S., & Leese, M. (2001). *Cluster analysis* (4th ed.). London: Arnold.
- Fabrellas, B., Ruiz, M. L., Martínez, M., Sanz, P., & Larrazábal, D. (2002). *Evaluación de la generación de dioxinas y furanos en el sector de la galvanización en caliente en el año 2002*.
- Falagán, A., Guerrero, J. E., & Serrano, A. (1995). Systèmes d'elevage caprin dans le sud de l'Espagne. In: *Goat production systems in the Mediterranean* (pp. 38-50). Wageningen, The Netherlands: Wageningen Pers.
- Falco, I., Cioppa, A., Iazzetta, A., Natale, P., & Tar, E. (1998). Optimizing neural networks for time series prediction. In *Proceedings of the Third World Conference on Soft Computing, (WSC3)*.
- Fang, S. C., Gao, D. Y., Shue, R. L., & Wu, S. Y. (2008). Canonical dual approach for solving 0-1 quadratic programming problems. *Journal of Industrial and Management Optimization*, 4(1), 125–142.
- Faraway, J., & Chatfield, C. (1998). Time series forecasting with neural networks: A case study. *Applied Statistics*, 47, 231–250. doi:10.1111/1467-9876.00109
- Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27, 861–874. doi:10.1016/j.patrec.2005.10.010
- Fawcett, T., & Niculescu-Mizil, A. (2007). PAV and the ROC convex hull. *Machine Learning*, 68(1), 97–106. doi:10.1007/s10994-007-5011-0
- Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). Knowledge discovery and data mining: Towards a unifying framework. In E. Simoudis, J. Han, & U. Fayyad (Eds.), *Proceeding of the Second International Conference on Knowledge Discovery and Data Mining (KDD'96)* (pp. 82-88). Menlo Park, CA: AAAI Press.
- Feldman, R., & Sanger, J. (2007). *The text mining handbook: Advanced approaches in analyzing unstructured data*. Cambridge, UK: Cambridge University Press.
- Fellbaum, C. (1998). *WordNet: An electronic lexical database* Cambridge, MA: The MIT Press.
- Feng, J., Haffner, P., & Gilbert, M. (2005). A learning approach to discovering Web page semantic structures. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 1055-1059). Washington, DC: IEEE Computer Society.
- Fenton, N. E., & Pleeger, S. L. (1997). *Software metrics: A rigorous & practical approach* (2nd ed.). Boston, MA: PWS.
- Ferilli, S., Basile, T. M. A., Di Mauro, N., Biba, M., & Esposito, F. (2007). Similarity-guided clause generalization. In R. Basili, & M. T. Pazienza (Eds.), *Proceedings of the AI*IA-2007: Artificial Intelligence and Human-Oriented Computing* (pp. 278-289). Berlin, Germany: Springer.
- Ferilli, S., Basile, T. M. A., Di Mauro, N., Biba, M., & Esposito, F. (2008). Generalization-based similarity for conceptual clustering. In Z. W. Ras, S. Tsumoto, & D. Zighed (Eds.), *Mining complex data* (pp. 13-26). Berlin, Germany: Springer.

- Fernández, C., Gómez, J., Sánchez Seiquer, P., Gómez-Chova, L., Soria, E., Moce, L., & Garcés, C. (2004). Prediction of weekly goat milk yield using autoregressive models. *South African Journal of Animal Science*, 34, 165–168.
- Fernández, C., Martínez, B., Gómez, E., Peris, C., Pascual, J. J., Serrano, A. J., & Soria, E. (2008a). Quantitative analysis of official milk control in Valencia community by SOM. In *Proceedings of the 9th International Conferences on Goats* (p. 106).
- Fernández, C., Pascual, J. J., Blas, E., & Cervera, C. (2008b). Milk prediction in dairy goats using multicomponent system model. *J. Appl. Anim. Res.*
- Fernández, C., Sánchez, A., & Garcés, C. (2002). Modeling the lactation curve for test-day milk yield in Murciano-Granadina goats. *Small Ruminant Research*, 46, 29–41. doi:10.1016/S0921-4488(02)00179-7
- Fernández, C., Soria, E., Mardalena, R., Martín, J. D., & Mata, C. (2007b). Qualitative analysis of feed management practice on goats herds by SOM in Murcia Region. *Journal of Applied Animal Research*, 32, 41–48.
- Fernández, C., Soria, E., Martín, J. D., & Serrano, A. J. (2006). Neural networks for animal science applications: Two case studies. *Expert Systems with Applications*, 31(2), 444–450. doi:10.1016/j.eswa.2005.09.086
- Fernández, C., Soria, E., Sánchez-Seiquer, P., Gómez-Chova, L., Magdalena, R., & Martín-Guerrero, J. D. (2007a). Weekly milk productions on dairy goats using neural networks. *Neural Computing and Application Journal*, 16, 373–381. doi:10.1007/s00521-006-0061-y
- Ferreira, P., & Azevedo, P. (2007). Evaluating deterministic motif significance measures in protein databases. *Algorithms for Molecular Biology; AMB*, 2(16).
- Ferri, C., Hernández-Orallo, J., & Modroiu, R. (2008). An experimental comparison of performance measures for classification. *Pattern Recognition Letters*, 30(1), 27–38. doi:10.1016/j.patrec.2008.08.010
- Figueiredo, M. A. T., & Jain, A. K. (2002). Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24, 381–396. doi:10.1109/34.990138
- Fikes, R. E., & Nilsson, N. J. (1971). STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2, 189–208. doi:10.1016/0004-3702(71)90010-5
- Fikes, R. E., Hart, P. E., & Nilsson, N. J. (1972). Learning and executing generalized robot plans. *Artificial Intelligence*, 3, 251–288. doi:10.1016/0004-3702(72)90051-3
- Firth, J. (1936). Alphabets and phonology in India and Burma. *Bulletin of the School of Oriental Studies, University of London*, 8, 517–546.
- Firth, J. (1957). A synopsis of linguistic theory 1930-1955. In *Studies in Linguistic Analysis* (pp. 1 -32). Oxford: Basil Blackwell.
- Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7, 179–188.
- Flach, P. A., & Matsubara, E. T. (2007). A simple lexicographic ranker and probability estimator. In *Proceedings of the 18th European Conference on Machine Learning* (pp. 575-582).
- Flach, P. A., & Wu, S. (2005). Repairing concavities in ROC curves. In *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 702-707).
- Flandrin, P. (1999). *Time-frequency/time-scale analysis*. San Diego, CA: Academic Press.
- Floratos, A. (1999). *Pattern discovery in biology: Theory and applications*. Unpublished doctoral dissertation, University of New York, USA.
- Floreano, D., & Mondada, F. (1996). Evolution of homing navigation in a real mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26, 396–407. doi:10.1109/3477.499791
- Floreano, D., & Urzelai, J. (2000). Evolutionary on-line self-organization of autonomous robots. In M. Sugisaka (Ed.), *Proceedings of the 5th International Symposium on Artificial Life and Robotics (AROB'2000)*, Oita, Japan.

Compilation of References

- Florenzano, D., & Urzelai, J. (2000). Evolutionary robotics: The next generation. In T. Gomi (Ed.), *Proceedings of Evolutionary Robotics III: From Intelligent Robots to Artificial Life*. Ontario, Canada: AAI Books.
- Florian, R., Ittycheriah, A., Jing, H., & Zhang, T. (2003). *Named entity recognition through classifier combination*. Paper presented at the Seventh Conference on Natural Language Learning (CoNLL 2003).
- Flury, B. (1990). Principal points. *Biometrika*, 77, 33–41. doi:10.1093/biomet/77.1.33
- Forgy, E. (1965). Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications. *Biometrics*, 21, 768–780.
- Fox, M., & Long, D. (1998). The automatic inference of state invariants in TIM. *Journal of Artificial Intelligence Research*, 9, 367–421.
- Fox, M., & Long, D. (2002). Extending the exploitation of symmetries in planning. In *Proceedings of the International Conference on AI Planning and Scheduling*.
- Fox, M., & Long, D. (2003). PDDL2.1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20, 61–124.
- Frank, C., Garg, A., Raheja, A., & Sztandera, L. (2003). Forecasting women's apparel sales using mathematical modeling. *International Journal of Clothing Science and Technology*, 15(2), 107–125. doi:10.1108/09556220310470097
- Frasconi, P., Gori, M., & Soda, G. (1992). Local feedback multilayered networks. *Neural Computation*, 4, 120–130. doi:10.1162/neco.1992.4.1.120
- Fréchet, M. (1948). Les éléments aléatoires de nature quelconque dans un espace distancié. *Ann. Inst. H. Poincaré*, 10, 215–310.
- Freifeld, C., Mandl, K., Reis, B., & Brownstein, J. (2008). HealthMap: Global infectious disease monitoring through automated classification and visualization of Internet media reports. *Journal of the American Medical Informatics Association*, 15, 150–157. doi:10.1197/jamia.M2544
- Freitag, D., & McCallum, A. (2000). *Information extraction with HMM structures learned by stochastic optimization*. Paper presented at the The Seventeenth National Conference on Artificial Intelligence.
- Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1), 119–139. doi:10.1006/jcss.1997.1504
- Friedman, J. H. (1991). Multivariate adaptive regression splines (with discussion). *Annals of Statistics*, 19, 1–141. doi:10.1214/aos/1176347963
- Friedman, N., & Koller, D. (2003). Being Bayesian about network structure. A Bayesian approach to structure discovery in Bayesian networks. *Machine Learning*, 50(1/2), 95–125. doi:10.1023/A:1020249912095
- Fritzke, B. (1994). Growing cells structures—a self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7(9), 1441–1460. doi:10.1016/0893-6080(94)90091-4
- Fritzke, B. (1995). A growing neural gas network learns topologies. In G. Tesuaro, D. Touretzky, & T. Leen, T. (Eds.), *Advances in neural information processing systems 7* (pp. 625–632). Cambridge, MA: MIT Press.
- Fritzke, B. (1997). *Some competitive learning methods* (draft document). Retrieved from <http://www.neuroinformatik.ruhr-uni-bochum.de/ini/VDM/research/gsn/JavaPaper>
- Fu, X., Ong, C. J., Keerthi, S., Hung, G. G., & Goh, L. (2004). Extracting the knowledge embedded in support vector machines. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN'04)*, Budapest.
- Fuart, F., Horby, D., & Best, C. (2007). *Disease outbreak surveillance through the Internet - the MediSys project*. Paper presented at the European Federation for Medical Informatics Special Topic Conference.
- Fuentetaja, R., & Borrajo, D. (2006). Improving control-knowledge acquisition for planning by active learning. In *Proceedings of the European Conference on Machine Learning*.

- Fukunaga, K., & Olsen, D. (1971). An algorithm for finding intrinsic dimensionality of data. *IEEE Transactions on Computers*, 20(2), 176–183. doi:10.1109/T-C.1971.223208
- Fukushima, K. (1975). Cognitron: A self-organizing multi-layered neural network. *Biological Cybernetics*, 20, 121–136. doi:10.1007/BF00342633
- Funahashi, K. (1989). On the approximate realization of continuous mapping by neural networks. *Neural Networks*, 2, 183–192. doi:10.1016/0893-6080(89)90003-8
- Fung, G., Sandilya, S., & Bharat, R. R. (2005). Rule extraction from linear support vector machines. In *KDD'05: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining* (pp. 32-40). New York: ACM Press.
- Futrelle, R. P., Shao, M., Cieslik, C., & Grimes, A. E. (2003). Extraction, layout analysis and classification of diagrams in PDF documents. In *Proceedings of the 7th International Conference on Document Analysis and Recognition* (pp. 1007-1014). Washington, DC: IEEE Computer Society.
- Gabriel, K. R., & Zamir, S. (1979). Lower rank approximation of matrices by least squares with any choices of weights. *Technometrics*, 21, 298–489. doi:10.2307/1268288
- Gabrilovich, E., & Markovitch, S. (2004). Text categorization with many redundant features: Using aggressive feature selection to make SVMs competitive with C4.5. In *Proceedings of the International Conference on Machine Learning ICML 2004*.
- Gabrilovich, E., & Markovitch, S. (2007). Computing semantic relatedness using Wikipedia-based explicit semantic analysis. Paper presented at the 20th International Joint Conference on Artificial Intelligence (IJCAI'07).
- Gabrys, B. (2000). Pattern classification for incomplete data. In *Proceedings of the International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies*, Brighton, UK (pp. 454-457).
- Gabrys, B. (2002). Neuro-fuzzy approach to processing inputs with missing values in pattern recognition problems. *International Journal of Approximate Reasoning*, 30(3), 149–179. doi:10.1016/S0888-613X(02)00070-1
- Galavotti, L., Sebastiani, F., & Simi, M. (2000). *Feature selection and negative evidence in automated text categorization*. Paper presented at the 4th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2000).
- Gales, M., & Young, S. (1993). *The theory of segmental hidden Markov models* (Tech. Rep. CUED/F-INFENG/TR 133), Cambridge University.
- Gallant, S. (1988). Connectionist expert systems. *Communications of the ACM*, 31(2), 152–169. doi:10.1145/42372.42377
- Ganchev, T., Parsopoulos, K. E., Vrahatis, M. N., & Fakotakis, N. (2008). Partially connected locally recurrent probabilistic neural networks. In X. Hu & P. Balasubramaniam (Eds.), *Recurrent neural networks* (pp. 377-400). Vienna, Austria: ARS Publishing.
- Ganchev, T., Tasoulis, D. K., Vrahatis, M. N., & Fakotakis, N. (2003, September). Locally recurrent probabilistic neural network for text-independent speaker verification. In *Proceedings of the 8th European Conference on Speech Communication and Technology, EUROSPEECH 2003* (Vol. 3, pp. 1673-1676).
- Ganchev, T., Tasoulis, D. K., Vrahatis, M. N., & Fakotakis, N. (2004). Locally recurrent probabilistic neural networks with application to speaker verification. *GESTS International Transaction on Speech Science and Engineering*, 1(2), 1–13.
- Ganchev, T., Tasoulis, D. K., Vrahatis, M. N., & Fakotakis, N. (2007). Generalized locally recurrent probabilistic neural networks with application to text-independent speaker verification. *Neurocomputing*, 70(7-9), 1424–1438. doi:10.1016/j.neucom.2006.05.012
- Gao, X.-Z., Ovaska, S. J., & Wang, X. (2008). A GA based negative selection algorithm. *International Journal of Innovative Computing . Information and Control*, 4(4), 971–979.

Compilation of References

- García-Hernández, J., Heras, S., Alfons, J., Paredes, R., Nácher, B., & Alemany, S. (2005). The morfo3D foot database. *Pattern Recognition and Image Analysis*, 3523, 658–665.
- García-Laencina, P. J., Figueiras-Vidal, A. R., Serrano-García, J., & Sancho-Gómez, J. L. (2005). Exploiting multitask learning schemes using private subnetworks. In J. Cabestany, et al. (Ed.), *Computational intelligence bioinspired systems* (LNCS 3512, pp. 233-240). Berlin, Germany: Springer.
- García-Laencina, P. J., Serrano, J., Figueiras-Vidal, A. R., & Sancho-Gómez, J. L. (2007). Multi-task neural networks for dealing with missing inputs. In J. Mira & J. R. Álvarez (Eds.), *Proceedings of the International Work Conference on the Interplay between Natural and Artificial Computation*, Murcia, Spain (LCNS 4527, pp. 282–291). Berlin, Germany: Springer.
- Garcia-Martinez, R., & Borrajo, D. (2000). An integrated approach of learning, planning, and execution. *Journal of Intelligent & Robotic Systems*, 29, 47–78. doi:10.1023/A:1008134010576
- Gartner, T., Driessens, K., & Ramon, J. (2003). Relational instance based regression for relational reinforcement learning. In *Proceedings of the International Conference on Machine Learning*.
- Gaweda, A. E., Muezzinoglu, M. K., Aronoff, G. R., Jacobs, A. A., Zurada, J. M., & Brier, M. E. (2005). Individualization of pharmacological anemia management using reinforcement learning. *Neural Networks*, 18, 826–834. doi:10.1016/j.neunet.2005.06.020
- Gaweda, A. E., Muezzinoglu, M. K., Aronoff, G. R., Jacobs, A. A., Zurada, J. M., & Brier, M. E. (2007). Using clinical information in goal-oriented learning for anemia management. *IEEE Engineering in Medicine and Biology Magazine*, 26(2), 27–36. doi:10.1109/EMB.2007.335580
- Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1), 1–58. doi:10.1162/neco.1992.4.1.1
- Ghahramani, Z. (1998). Learning dynamic Bayesian networks. In C. L. Giles & M. Gori (Eds.), *Adaptive processing of sequences and data structures* (pp. 168–197). Berlin, Germany: Springer-Verlag.
- Ghahramani, Z. (2002). Graphical models: Parameter learning. In M. A. Arbib (Ed.), *The Handbook of brain theory and neural networks*. Cambridge, MA: MIT Press.
- Ghahramani, Z., & Beal, M. J. (2000). Variational inference for Bayesian mixtures of factor analysers. In Solla, Leen, & Müller (eds), *Advances in Neural Information Processing Systems 12*, MIT Press, 449-455.
- Ghahramani, Z., & Jordan, M. I. (1994). *Learning from incomplete data* (Tech. Rep. AIM-1509). Massachusetts Institute of Technology, Cambridge, MA, USA.
- Ghahramani, Z., & Jordan, M. I. (1994). Supervised learning from incomplete data via an EM approach. In J. D. Cowan, et al. (Ed.), *Advances on neural information processing systems 6*. (pp. 120-127). San Francisco: Morgan Kaufmann Publishers Inc.
- Ghallab, M., Nau, D., & Traverso, P. (2004). *Automated task planning. Theory & practice*. San Francisco: Morgan Kaufmann.
- Ghosh, J., & Shin, Y. (1991). The pi-sigma networks: An efficient higher-order neural network for pattern classification and function approximation. In *Proceedings of the International Joint Conference on Neural Networks*, Seattle, Washington (pp. 13-18).
- Gibson, R. (2007). *Radio and television reporting*. Retrieved from http://www.fdv.uni-lj.si/Predmeti/Diplomski/Arhiv/ucni_nacrti_2002-03.pdf
- Gil, Y. (1992). *Acquiring domain knowledge for planning by experimentation*. Unpublished doctoral dissertation, Carnegie Mellon University.
- Gilb, T., & Graham, D. (1993). *Software inspection*. Reading MA: Addison-Wesley.
- Gipson, T. A., & Grossman, M. (1990). Lactation curves in dairy goats: A review. *Small Ruminant Research*, 3, 383. doi:10.1016/0921-4488(90)90019-3

- Girolami, M. (2002). Latent variable models for the topographic organisation of discrete and strictly positive data. *Neurocomputing*, 48, 185–198. doi:10.1016/S0925-2312(01)00659-2
- Girosi, F., & Poggio, T. (1990). Networks and the best approximation property. *Biological Cybernetics*, 63(3), 169–176.
- Gökberk, B., Salah, A. A., & Akarun, L. (2005). Rank-based decision fusion for 3D shape-based face recognition. In *Proc. of the Int. Conf. Audio- and Video-based Biometric Person Authentication* (LNCS 3546, pp. 1019–1028).
- Gökberk, B., Salah, A. A., Alyüz, N., & Akarun, L. (in press). 3D face recognition: Technology and applications. In M. Tistarelli, S. Z. Li, & R. Chellappa (Eds.), *Biometrics for surveillance and security*. London: Springer-Verlag.
- Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA: Addison-Wesley.
- Goldberg, D., & Sastry, K. (2007). *Genetic algorithms: The design of innovation*. Berlin, Germany: Springer.
- González-Marcos, A., Ordieres-Meré, J. B., Pernía-Espinoza, A. V., & Torre-Suárez, V. (2008). Desarrollo de un cerrojo artificial para el skin-pass en una línea de acero galvanizado por inmersión en caliente. *Revista de Metalurgia*, 44(1), 29–38. doi:10.3989/revmetalm.2008.v44.i1.93
- Good, I. J. (1952). Rational decisions. *Journal of the Royal Statistical Society. Series B. Methodological*, 14, 107–114.
- Good, I. J. (1968). Corroboration, explanation, evolving probability, simplicity, and a sharpened razor. *The British Journal for the Philosophy of Science*, 19, 123–143. doi:10.1093/bjps/19.2.123
- Gorban, A. N., & Rossiev, A. A. (1999). Neural network iterative method of principal curves for data with gaps. *Journal of Computer and Systems Sciences International*, 38(5), 825–830.
- Gorban, A. N., Pitenco, A. A., Zinov'ev, A. Y., & Wunsch, D. C. (2001). Visualization of any data using elastic map method. *Smart Engineering System Design*, 11, 363–368.
- Gorban, A. N., Zinovyev, A. Y., & Wunsch, D. C. (2003). Application of the method of elastic maps in analysis of genetic texts. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN*, Portland, Oregon.
- Gorban, A., & Zinovyev, A. (2005). Elastic principal graphs and manifolds and their practical applications. *Computing*, 75, 359–379. doi:10.1007/s00607-005-0122-6
- Gorban, A., & Zinovyev, A. (2008). Elastic maps and nets for approximating principal manifolds and their application to microarray data visualization. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 96–130). Berlin, Germany: Springer.
- Gorban, A., & Zinovyev, A. (2008). PCA and K-means decipher genome. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 309–323). Berlin, Germnay: Springer.
- Gorban, A., Kégl, B., Wunsch, D., & Zinovyev, A. (Eds.). (2008). *Principal manifolds for data visualization and dimension reduction*. Berlin, Germany: Springer.
- Gorban, A., Sumner, N. R., & Zinovyev, A. (2008). Beyond the concept of manifolds: Principal trees, metro maps, and elastic cubic complexes. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 219–237). Berlin, Germany: Springer.
- Gorban, A., Sumner, N., & Zinovyev, A. (2007). Topological grammars for data approximation. *Applied Mathematics Letters*, 20(4), 382–386. doi:10.1016/j.aml.2006.04.022
- Gori, M. (1989, November). An extension of BPS. In *Proceedings of the 2nd International Workshop on Neu-*

Compilation of References

- ral Networks and their Applications*, Nimes, France (pp. 83-93).
- Gori, M., & Soda, G. (1990). Temporal pattern recognition using EBPS. In *Proceedings of the EURASIP Workshop 1990 on Neural Networks* (LNCS 412, pp. 69-80). London, UK: Springer-Verlag.
- Gori, M., Bengio, Y., & de Mori, R. (1989). BPS: A learning algorithm for capturing the dynamic nature of speech. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN'89* (Vol. 2, pp. 417-423).
- Gorse, D., Shepherd, A. J., & Taylor, J. G. (1997). The new ERA in supervised learning. *Neural Networks*, 10(2), 343–352. doi:10.1016/S0893-6080(96)00090-1
- Gowrishankar, , & Satyanarayana, P. S. (2008). Recurrent neural network based bit error rate prediction for 802.11 wireless local area network. *International Journal of Computer Sciences and Engineering Systems*, 2(3), 177-184.
- Granitto, P. M., Verdes, P. F., & Ceccatto, H. A. (2005). Large-scale investigation of weed seed identification by machine vision. *Computers and Electronics in Agriculture*, 47, 15–24. doi:10.1016/j.compag.2004.10.003
- Grassi, G., & Cafagna, D. (2003). Locally-connected neural networks: Stability analysis and synthesis technique. In N. Mastorakis (Ed.), *Recent advances in intelligent systems and signal processing* (pp. 321-326).
- Gray, A. R. (1999). A simulation-based comparison of empirical modeling techniques for software metric models of development effort. In *Proceedings of the International Conference on Neural Information Processing* (pp. 526-531). Washington, DC: IEEE.
- Gray, A., & MacDonell, S. (1997). Applications of fuzzy logic to software metric models for development effort estimation. In *Proceedings of the 1997 Annual meeting of the North American Fuzzy Information Proceeding Society- NAFIPS'97* (pp. 394-399). Washington, DC: IEEE.
- Grenader, T., Klein, D., & Manning, C. (2005). *Unsupervised learning of field segmentation models for information extraction*. Paper presented at the 43rd Annual Meeting on Association for Computational Linguistics.
- Grinberg, D., Lafferty, J., & Sleator, D. (1995). *A robust parsing algorithm for link grammars*. Paper presented at the Fourth International Workshop on Parsing Technologies.
- Gross, R., Matthews, I., Cohn, J., Kanade, T., & Baker, S. (2008). Multi-PIE. In *Proc. of the 8th IEEE Conf. on Automatic Face and Gesture Recognition*, Amsterdam.
- Grossberg, S. (1976). Adaptive pattern classification and universal recoding: I. Parallel development and coding of neural feature detectors. *Biological Cybernetics*, 23, 121–134. doi:10.1007/BF00344744
- Grossberg, S. (1976). Adaptive pattern recognition and universal encoding: II. Feedback, expectation, olfaction, and illusions. *Biological Cybernetics*, 23, 187–202. doi:10.1007/BF00344744
- Grossberg, S. (1978). A theory of human memory: Self-organization and performance of sensory-motor codes, maps, and plans. In R. Rosen & F. Snell (Eds.), *Progress in theoretical biology* (Vol. 5). New York: Academic Press.
- Grzesiak, W., Blaszczyk, P., & Lacroix, R. (2006). Methods of predicting milk yield in dairy cow; predictive capabilities of Wood's lactation curve and artificial neural network. *Computers and Electronics in Agriculture*, 54, 69–83. doi:10.1016/j.compag.2006.08.004
- Guerra, F. A., & Coelho, L. S. (2008). Multi-step ahead nonlinear identification of Lorenz's chaotic system using radial basis neural network with learning by clustering and particle swarm optimization . *Chaos, Solitons, and Fractals*, 35(5), 967–979.
- Guha, S., Rastogi, R., & Shim, K. (1998). CURE: An efficient clustering algorithm for large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (pp. 73-84).
- Guha, S., Rastogi, R., & Shim, K. (2000). ROCK: A robust clustering algorithm for categorical attributes. *Information Systems*, 25(5), 345–366. doi:10.1016/S0306-4379(00)00022-3

- Gui, M., Das, S., & Pahwa, A. (2007). Procreating v-detectors for nonself recognition: An application to anomaly detection in power systems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, London, UK (pp. 261-268).
- Guo, G., & Shouyi, Y. (2003). Evolutionary parallel local search for function optimization. *IEEE Transactions on Systems . Man and Cybernetics Part-B*, 7(1), 243–258.
- Guo, H.-F., Mahmud, J., Borodin, Y., Stent, A., & Ramakrishnan, I. V. (2007). A general approach for partitioning Web page content based on geometric and style information. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 929-933). Washington, DC: IEEE Computer Society.
- Guo, Z., & Uhrig, R. (1992). Using genetic algorithm to select inputs for neural networks. In *Proceedings of the Workshop on Combinations of Genetic Algorithms and Neural Networks, COGANN92* (pp. 223-234).
- Gupta, A., & Lam, M. S. (1996). Estimating missing values using neural networks. *The Journal of the Operational Research Society*, 47, 229–238.
- Gusfield, D. (1999). *Algorithms on strings, trees and sequences: Computer science d computational biology*. Cambridge, UK: Cambridge University Press.
- Hadjar, K., Rigamonti, M., Lalanne, D., & Ingold, R. (2004). Xed: A new tool for extracting hidden structures from electronic documents. In *Proceedings of the 1st International Workshop on Document Image Analysis for Libraries* (p. 212). Washington, DC: IEEE Computer Society.
- Halatchev, M., & Gruenwald, L. (2005). Estimating missing values in related sensor data streams. In *Proceedings of the International Conference on Management of Data*, Goa, India (pp. 83-94).
- Hamilton, J. D. (1994). *Time series analysis: An applied focus with emphasis on economics and assumption reader has an economics background*. Princeton, NJ: Princeton U. Press.
- Hammond, J. H. (1990). *Quick response in the apparel industries*. Cambridge, MA: Harvard Business School.
- Hammouda, K., & Kamel, M. (2004). Efficient phrase-based document indexing for Web document clustering. *IEEE Transactions on Knowledge and Data Engineering*, 16(10), 1279–1296. doi:10.1109/TKDE.2004.58
- Hamouz, M., Kittler, J., Kamarainen, J. K., Paalanen, P., Kalviainen, H., & Matas, J. (2005). Feature-based affine-invariant localization of faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(9), 1490–1495. doi:10.1109/TPAMI.2005.179
- Hamza, H., Belaïd, Y., & Belaïd, A. (2007). A case-based reasoning approach for invoice structure extraction. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 327-331). Washington, DC: IEEE Computer Society.
- Han, J., & Kamber, M. (2001). *Data mining concepts and techniques*. San Francisco: Morgan Kaufmann.
- Hand, D. J., Mannila, H., & Smyth, P. (2001). *Principles of data mining*. Cambridge, MA: MIT Press.
- Hanke, J. E., Wichern, D. W., & Reitsch, A. G. (2001). *Business forecasting* (7th ed.). Upper Saddle River, NJ: Prentice Hall.
- Hansen, J., McDonald, J., & Slice, J. (1992). Artificial intelligence and generalized qualitative response models: An empirical test on two audit decision-making domains . *Decision Sciences*, 23, 708–723. doi:10.1111/j.1540-5915.1992.tb00413.x
- Hansen, P., & Jaumard, B. (1997). Cluster analysis and mathematical programming. *Mathematical Programming*, 79, 191–215.
- Happel, B. L. M., & Murre, M. J. (1994). The design and evolution of modular neural network architectures. *Neural Networks*, 7, 985–1004. doi:10.1016/S0893-6080(05)80155-8
- Hara, K. (2006). *Applying a SVM based chunker and a text classifier to the Deid challenge*. Paper presented at the i2b2 Workshop on Challenges in Natural Language Processing for Clinical Data.
- Harmer, P. K., Williams, P. D., Gunsch, G. H., & Lamont, G. B. (2002). An artificial immune system architecture

Compilation of References

- for computer security applications. *IEEE Transactions on Evolutionary Computation*, 6(3), 252–280. doi:10.1109/TEVC.2002.1011540
- Harnett, J. D., Foley, R. N., Kent, G. M., Barre, P. E., Murray, D., & Parfrey, P. S. (1995). Congestive heart failure in dialysis patients: Prevalence, incidence, prognosis and risk factors. *Kidney International*, 47, 884–890. doi:10.1038/ki.1995.132
- Harris, Z. (1968). *Mathematical structures of language*. New York: Interscience Publishers.
- Hartigan, J. (1975). *Clustering algorithms*. New York: John Wiley & Sons.
- Hartman, E. J., Keeler, J. D., & Kowalski, J. M. (1990). Layered neural networks with Gaussian hidden units as universal approximations . *Neural Computation*, 2, 210–215.
- Hastie, T. (1984). *Principal curves and surfaces*. Unpublished doctoral dissertation, Stanford University, CA.
- Hatzakis, G., & Tsoukas, C. (2001). Neural networks in the assessment of HIV immunopathology. In *Proceedings of the AMIA Symposium* (pp. 249-53).
- Hayashi, Y., & Imura, A. (1990). Fuzzy neural expert system with automated extraction of fuzzy If-then rules from a trained neural network. In *Proceedings of First International Symposium on Uncertainty Modeling and Analysis* (pp. 489-494).
- Haykin, S. (1999). *Neural networks: A comprehensive foundation*. Upper Saddle River, NJ: Prentice Hall.
- Haykin, S. (2008). *Neural networks: A comprehensive foundation* (3rd ed.). New York: Macmillan.
- Hecht, F., & Shiel, W. (2003). *Webster's new world medical dictionary* (2nd ed.). New York: John Wiley & Sons Publishing.
- Heinen, T. (1996). Latent class and discrete latent trait models: Similarities and differences, Housand Oaks, California: Sage.
- Heitz, C. (1995). Optimum time-frequency representations for the classification and detection of signals. *Applied Signal Processing*, 3, 124–143.
- Henikoff, S., & Henikoff, J. (1993). Performance evaluation of amino acid substitution matrices. *Proteins*, 17(1), 49–61. doi:10.1002/prot.340170108
- Hernando, M. E., Gomez, E. J., del Pozo, F., & Corcoy, R. (1996). DIABNET: A qualitative model-based advisory system for therapy planning in gestational diabetes. *Medical Informatics*, 21(4), 359–374.
- Hill, T., Marquez, L., O'Connor, M., & Remus, W. (1994). Artificial neural network models for forecasting and decision making. *International Journal of Forecasting*, 10(1), 5–15. doi:10.1016/0169-2070(94)90045-0
- Hinton, G. E., & Zemel, R. S. (1994). Autoencoders, minimum description length and Helmholtz free energy, In Cowan, Tesauro, & Alspector (eds), *Advances in Neural Information Processing Systems 6*, Morgan Kaufmann Pub., 449-455.
- Hinton, G. E., Dayan, P., Frey, B. J., & Neal, R. N. (1995). The wake-sleep algorithm for unsupervised learning neural networks . *Science*, 268, 1158–1160.
- Hinton, G., Osindero, S., & Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Computation*, 18, 1527–1554. doi:10.1162/neco.2006.18.7.1527
- Hoffmann, J., & Nebel, B. (2001). The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14, 253–302.
- Hogg, C., Munoz-Avila, H., & Kuter, U. (2008). HTN-MAKER: Learning HTNs with Minimal Additional Knowledge Engineering Required. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*.
- Holewski, J. J., Moss, K. M., Stess, R. M., Graf, P. M., & Grunfeld, C. (1989). Prevalence of foot pathology and lower extremity complications in a diabetic outpatient clinic. *Journal of Rehabilitation Research and Development*, 26, 35–44.
- Holland, J. H. (1973). Genetic algorithms and the optimal allocation of trials. *SIAM Journal on Computing*, 2(2), 88–105. doi:10.1137/0202009
- Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, 11, 63–91. doi:10.1023/A:1022631118932

- Holzreiter, S. H., & Köle, M. E. (1993). Assessment of gait patterns using neural networks. *Journal of Biomechanics*, 26(6), 645–651. doi:10.1016/0021-9290(93)90028-D
- Honeine, P., & Richard, C. (2007). Signal-dependent time-frequency representations for classification using a radially gaussian kernel and the alignment criterion. In *Proc. of the IEEE Statistical Signal Processing Workshop*, Madison, WI, USA.
- Honeine, P., Richard, C., & Flandrin, P. (2007). Time-frequency kernel machines. *IEEE Transactions on Signal Processing*, 55, 3930–3936. doi:10.1109/TSP.2007.894252
- Honeine, P., Richard, C., Flandrin, P., & Pothin, J.-B. (2006). Optimal selection of time-frequency representations for signal classification: a kernel-target alignment approach. In *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Toulouse, France.
- Honkela, T., Kaski, S., Lagus, K., & Kohonen, T. (1997). WEBSOM--self-organizing maps of document collections. In *Proceedings of the Workshop on Self-Organizing Maps – WSOM'97*, Helsinki University of Technology, Neural Networks Research Center, Espoo, Finland (pp. 310-315).
- Horchler, A., Reeve, R., Webb, B., & Quinn, R. (2004). Robot phonotaxis in the wild: A biologically inspired approach to outdoor sound localization. *Journal in Advanced Robotics*, 18(8), 801–816. doi:10.1163/1568553041738095
- Hornik, K., Stinchcombe, M., & White, H. (1989). Multi-layer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359–366. doi:10.1016/0893-6080(89)90020-8
- Hotelling, H. (1933). Analisys of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24, 417–441. doi:10.1037/h0071325
- Hu, M., & Liu, B. (2004). *Mining and summarizing customer reviews*. Paper presented at the International Conference on Knowledge Discovery and Data Mining (KDD'04).
- Hu, Y. C., & Tseng, F. M. (2007). Functional-link net with fuzzy integral for bankruptcy prediction. *Neurocomputing*, 70, 2959–2968. doi:10.1016/j.neucom.2006.10.111
- Huang, J., & Douglas, B. (2001). The eMOTIF database. *Nucleic Acids Research*, 29(1), 202–204. doi:10.1093/nar/29.1.202
- Huang, J., Georgiopoulos, M., & Heileman, G. (1995). Fuzzy ART properties. *Neural Networks*, 8(2), 203–213. doi:10.1016/0893-6080(94)00073-U
- Huang, S., Sun, J.-T., Wang, X., Zeng, H.-J., & Chen, Z. (2006). *Subjectivity categorization of Weblog with part-of-speech based smoothing*. Paper presented at the Sixth International Conference on Data Mining (ICDM'06).
- Huang, W., Xu, B., & Chan-Hilton, A. (2004). Forecasting flows in Apalachicola River using neural networks. *Hydrological Processes*, 18, 2545–2564. doi:10.1002/hyp.1492
- Huang, X., Acero, A., & Hon, H. (2001). *Spoken language processing*. Upper Saddle River, NJ: Prentice Hall.
- Huffman, S. B., Pearson, D. J., & Laird, J. E. (1992). Correcting imperfect domain theories: A knowledge level analysis. In *Machine learning: Induction, analogy and discovery*.
- Hughes, R. T. (1996). *An evaluation of machine learning techniques for software effort estimation*. University of Brighton.
- Hulo, N., Bairoch, A., Bulliard, V., Cerutti, L., De Castro, E., & Langendijk-Genevaux, P. (2006). The PROSITE database. *Nucleic Acids Research*, 34, D227–D230. doi:10.1093/nar/gkj063
- Hung, K.-k., Cheung, Y.-m., & Xu, L. (2003). An extended ASLD trading system to enhance portfolio management. *IEEE Transactions on Neural Networks*, 14(2), 413–425. doi:10.1109/TNN.2003.809423
- Hunter, L. (1993). Molecular biology for computer scientists. In L. Hunter (Ed.), *Artificial intelligence and molecular biology* (pp. 1-46). Menlo Park, CA: AAAI Press.

Compilation of References

- Idri, A., Khosgoftaar, T. M., & Abran, A. (2002). Can neural networks be easily interpreted in software cost estimation? In *Proceedings of the 2002 World Congress on Computational Intelligence*, Honolulu, Hawaii (pp. 12-17).
- Ilghami, O., Nau, D., & Héctor Muñoz-Avila, H. (2006). Learning to do HTN planning. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling*.
- Ilghami, O., Nau, D., Muñoz-Avila, H., & Aha, D. W. (2002). CaMel: Learning method preconditions for HTN planning. In *Proceedings of the 6th International Conference on Artificial Intelligence Planning Systems (AIPS02)*.
- Inoue, T., & Abe, S. (2001). *Fuzzy Support vector machines for pattern classification*. In *Proceedings of the IJCNN'01, International Joint Conference on Neural Network*, 2 (pp. 1449-1454).
- Institute for parallel and distributed high performance systems. (n.d.). *SNNS – Stuttgart neural network simulator. User manual, ver. 4.2* [online]. Retrieved from <http://www-ra.informatik.uni-tuebingen.de/downloads/SNNS/SNNSSv4.2.Manual.pdf>
- Isaksson, M., Wisell, D., & Ronnow, D. (2005). Wide-band dynamic modeling of power amplifiers using radial-basis function neural networks. *IEEE Transactions on Microwave Theory and Techniques*, 53(11), 3422–3428.
- Ishibuchi, H., Miyazaki, A., Kwon, K., & Tanaka, H. (1993). Learning from incomplete training data with missing values and medical application. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, Nagoya, Japan (pp. 1871-1874).
- Ishibuchi, H., Murata, T., & Turksen, I. B. (1997). Single-objective and two-objective genetic algorithms for selecting linguistics rules for pattern classification problems. *Fuzzy Sets and Systems*, 89, 135–149. doi:10.1016/S0165-0114(96)00098-X
- Ishibuchi, H., Nakashima, T., & Murata, T. (1999). Performance evaluation of fuzzy classifier systems for multidimensional pattern classification problems. *IEEE Transactions on Systems, Man, and Cybernetics*, 29(5), 601–618. doi:10.1109/3477.790443
- Ishibuchi, H., Nozaki, K., & Tanaka, H. (1992). Distributed representation of fuzzy rules and its application to pattern classification. *Fuzzy Sets and Systems*, 52, 21–32. doi:10.1016/0165-0114(92)90032-Y
- Ishibuchi, H., Nozaki, K., Yamamoto, N., & Tanaka, H. (1995). Selecting fuzzy if-then rules for classification problem using genetic algorithms. *IEEE transactions on Fuzzy Systems*, 3, 260–270. doi:10.1109/91.413232
- Ivakhno, S., & Armstrong, J. D. (2007). Non-linear dimensionality reduction of signalling networks. *BMC Systems Biology*, 1(27).
- Jaakkola, T. S. (2001), Tutorial on variational approximation methods, in Opper & Saad (eds), *Advanced Mean Field Methods: Theory and Practice*, MIT press, 129-160.
- Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts . *Neural Computation*, 3, 79–87.
- Jain, A. K., Dass, S. C., & Nandakumar, K. (2004). Soft biometric traits for personal recognition systems. In *Proc. of the Int. Conf. Biometric Authentication*, Hong-Kong.
- Jain, A. K., Duin, R. P. W., & Mao, J. (2000). Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), 4–37. doi:10.1109/34.824819
- Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: A review. *ACM Computing Surveys*, 31(3), 264–323. doi:10.1145/331499.331504
- Jain, A., & Dubes, R. (1988). *Algorithms for clustering data*. Englewood Cliffs, NJ: Prentice Hall.
- Jain, A., Murty, M., & Flynn, P. (1999). Data clustering: A review. *ACM Computing Surveys*, 31(3), 264–323. doi:10.1145/331499.331504
- James, M. (1985). *Classification algorithms*. London: Ed. Collins.

- Jayawardhana, B., Xie, L., & Yuan, S. (2002, August). Active control of sound based on diagonal recurrent neural network. In *Proceedings of the Conference of the Society of Instrument and Control Engineers (SICE 2002)*, Osaka, Japan (pp. 2666-2671).
- Jerez, J. M., Molina, I., Subirats, J. L., & Franco, L. (2006). Missing data imputation in breast cancer prognosis. In *Proceedings of the IASTED International Multiconference in Biomedical Engineering*, Innsbruck, Austria (pp. 323-328).
- Ji, C., & Elwalid, A. (2000). Measurement-based network monitoring: missing data formulation and scalability analysis. In *Proceedings of the IEEE International Symposium on Information Theory*, Sorrento, Italy (pp. 78).
- Ji, H., & Grisham, R. (2008). *Refining event extraction through unsupervised cross-document inference*. Paper presented at the Annual Meeting of the Association of Computational Linguistics.
- Ji, Z. (2005). A boundary-aware negative selection algorithm. In *Proceedings of the 9th IASTED International conference on Artificial intelligence and soft computing (ASC' 05)*, Benidorm, Spain.
- Ji, Z., & Dasgupta, D. (2004). Real valued negative selection algorithm using variable sized detectors. In *Proceedings of the Genetic and Evolutionary Computation Conference*, Seattle, WA (pp. 287-298).
- Ji, Z., & Dasgupta, D. (2006). Applicability issues of the real-valued negative selection algorithms. In *Proceedings of the Genetic and Evolutionary Computation Conference*, Seattle, WA (pp. 111-118).
- Ji, Z., & Dasgupta, D. (2007). Revisiting negative selection algorithms. *Evolutionary Computation Journal*, 15(2), 223–251. doi:10.1162/evco.2007.15.2.223
- Jian, K., Chen, H., & Yuan, S. (2005). Classification for incomplete data using classifier ensembles. In *Proceedings of the International Conference on Neural Networks and Brain*, Beijing, China (pp. 559-563).
- Jiménez, S., Fernández, F., & Borrajo, D. (2008). The PELA architecture: Integrating planning and learning to improve execution. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*.
- Jindal, N., & Liu, B. (2008). *Opinion spam and analysis*. Paper presented at the International Conference on Web Search and Web Data Mining.
- Jo, H., & Han, I. (1996). Integration of case-based forecasting, neural network, and discriminant analysis for bankruptcy prediction. *Expert Systems with Applications*, 11(4), 415–422. doi:10.1016/S0957-4174(96)00056-5
- Joachims, T. (2002). *Learning to classify text using support vector machines - methods, theory, and algorithms*. Amsterdam: Kluwer Academic Publishers.
- John, R., & Holm, J. R. (1995). *Elements of general, organic and biological chemistry* (9th ed.). New York: John Wiley and Sons.
- Jolliffe, I. T. (2002). *Principal component analysis, series: Springer series in statistics, 2nd ed., XXIX*. New York: Springer.
- Jonassen, I. (2000). Methods for discovering conserved patterns in protein sequences and structures. In D. Higgins, & W. Taylor (Eds.), *Bioinformatics: Sequence, structure and databanks. A practical approach*. Oxford, UK: Oxford University Press.
- Jonassen, I., Collins, J., & Higgins, D. (1995). Finding flexible patterns in unaligned protein sequences. *Protein Science*, 4(8), 1587–1595. doi:10.1002/pro.5560040817
- Jones, A. J. (1993). Genetic algorithms and their applications to the design of neural networks. *Neural Computing & Applications*, 1, 32–45. doi:10.1007/BF01411373
- Jones, C., & Purves, R. (2008). Geographical information retrieval. *International Journal of Geographical Information Science*, 22(3), 219–228. doi:10.1080/13658810701626343
- Jordan, C., & Bartlett, R. (1995). Pressure distribution and perceived comfort in casual footwear. *Gait & Posture*, 3, 215–220. doi:10.1016/0966-6362(96)82850-5
- Jordan, M. I., & Jacobs, R. A. (1994). Hierarchical mixtures of experts and the EM algorithm. *Neural Computation*, 6, 181–214.

Compilation of References

- Jordan, M. I., & Xu, L. (1995). Convergence Results for The EM Approach to Mixtures of Experts Architectures. *Neural Networks*, 8, 1409–1431.
- Jordan, M., Ghahramani, Z., Jaakkola, T., & Saul, L. (1999). Introduction to variational methods for graphical models . *Machine Learning*, 37, 183–233.
- Jorgensen, M. (1995). Experience with accuracy of software maintenance task effort prediction models. *IEEE Transactions on Software Engineering*, 21(8), 674–681. doi:10.1109/32.403791
- Jurafsky, D., & Martin, J. (2008). *Speech and language processing: An introduction to natural language processing, computational linguistics and speech recognition* (2nd ed.). Upper Saddle River, NJ: Prentice Hall.
- Juszczak, P., & Duin, R. P. W. (2004). Combining one-class classifiers to classify missing data. In F. Roli, et al. (Ed.), *Multiple classifier systems* (LNCS 3077, pp. 92-101). Berlin, Germany: Springer.
- Kabán, A., & Girolami, M. (2002). A dynamic probabilistic model to visualise topic evolution in text streams. *Journal of Intelligent Information Systems*, 18(2/3), 107–125. doi:10.1023/A:1013673310093
- Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4, 237–285.
- Kagehiro, T., & Fujisawa, H. (2008). Multiple hypotheses document analysis. In S. Marinai & H. Fujisawa (Eds.), *Machine learning in document analysis and recognition* (pp. 1-24). Berlin, Germany: Springer.
- Kakas, A. C., & Mancarella, P. (1990). On the relation of truth maintenance and abduction. In *Proceedings of the 1st Pacific Rim International Conference on Artificial Intelligence*, Nagoya, Japan.
- Kambhatla, N., & Leen, T. K. (1997). Dimension reduction by local PCA. *Neural Computation*, 9, 1493–1516. doi:10.1162/neco.1997.9.7.1493
- Kan, S. (1995). *Metrics and models in software quality engineering*. Reading, MA: Addison-Wesley.
- Kandola, J., Shawe-Taylor, J., & Cristianini, N. (2002). *Optimizing kernel alignment over combinations of kernels*. London: University of London, Department of Computer Science.
- Karaboğa, D., & Ökdem, S. (2004). A simple and global optimization algorithm for engineering problems: differential evolution algorithm. *Turk J Elec Engin*, 12(1).
- Karami, A., & Mohammadi, M. S. (2008). Radial basis function neural network for power system load-flow . *International Journal of Electrical Power & Energy Systems*, 30(1), 60–66.
- Kardirkamanathan, V., Niranjan, M., & Fallside, F. (1991), Sequential adaptation of Radial basis function neural networks and its application to time-series prediction, In Lippmann, Moody, & Touretzky (eds), *Advances in Neural Information Processing System 3*, Morgan Kaufmann Pub., 721-727.
- Karels, G. V., & Prakash, A. J. (1987). Multivariate normality and forecasting for business bankruptcy. *Journal of Business Finance & Accounting*, 14, 573–593. doi:10.1111/j.1468-5957.1987.tb00113.x
- Karhunen, K. (1946). Zur spektraltheorie stochastischer prozesse. *Suomalainen Tiedeakatemia Toimituksia. Sar. A.4: Biologica*, 37.
- Karthikchandra, D., Ravi, V., & Bose, I. (in press). Failure prediction of dotcom companies using hybrid intelligent techniques. *Expert Systems with Applications*. doi:.doi:10.1016/j.eswa.2008.05.047
- Karunanithi, N., Whitley, D., & Malaiya, Y. K. (1992). Prediction of software reliability using connectionist models. *IEEE Transactions on Software Engineering*, 18(7), 563–574. doi:10.1109/32.148475
- Kasabov, N. K., & Song, Q. (2002). DENFIS: Dynamic evolving neural-fuzzy inference system and its application for time-series prediction. *IEEE transactions on Fuzzy Systems*, 10(2), 144–154. doi:10.1109/91.995117
- Kaski, S., Honkela, T., Lagus, K., & Kohonen, T. (1998). WEBSOM – self-organizing maps of document collections. *Neurocomputing*, 21, 101–117. doi:10.1016/S0925-2312(98)00039-3

- Kasper, K., Reininger, H., & Wust, H. (1996). Strategies for reducing the complexity of a RNN-based speech recognizer. In *Proceedings of the IEEE Conference on Acoustics, Speech, and Signal Processing*, 6, 3354–3357. doi:10.1109/ICASSP.1996.550596
- Kasper, K., Reininger, H., Wolf, D., & Wust, H. (1995). A speech recognizer based on locally recurrent neural networks. In *Proceedings of the International Conference on Artificial Neural Networks*, 2, 15–20.
- Kazemy, A., Hosseini, S. A., & Farrokhi, M. (2007, June). Second order diagonal recurrent neural network. In *Proceedings of the IEEE International Symposium on Industrial Electronics, ISIE, 2007*, 251–256.
- Kearns, M. J., & Vazirani, U. V. (1994). *An introduction to computational learning theory*. Cambridge, MA: MIT Press.
- KEEL. (2009). *KEEL: Knowledge extraction based on evolutionary learning*. Retrieved from <http://sci2s.ugr.es/keel/index.php>
- Kegl, B. (1999). *Principal curves: Learning, design, and applications*. Unpublished doctoral dissertation, Concordia University, Canada.
- Kegl, B., & Krzyzak, A. (2002). Piecewise linear skeletonization using principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1), 59–74. doi:10.1109/34.982884
- Keller, R. (1987). *The role of explicit contextual knowledge in learning concepts to improve performance* (Tech. Rep.). Rutgers University.
- Kennedy, A., & Szpakowicz, S. (2008). *Evaluating Roget's thesauri*. Paper presented at the Association for Computational Linguistics (ACL 2008).
- Keogh, E., & Lin, J. (2005). Clustering of time series subsequences is meaningless: Implications for previous and future research. *Knowledge and Information Systems*, 8(2), 154–177. doi:10.1007/s10115-004-0172-7
- Khardon, R. (1999). Learning action strategies for planning domains. *Artificial Intelligence*, 113, 125–148. doi:10.1016/S0004-3702(99)00060-0
- Khoshgoftaar, T. M., & Szabo, R. M. (1996). Using neural networks to predict software faults during testing. *IEEE Transactions on Reliability*, 45(3), 456–462. doi:10.1109/24.537016
- Khoshgoftaar, T. M., Pandya, A. S., & More, H. B. (1992). A neural network approach for predicting software development faults. In *Proceeding of the 3rd International Symposium on Software Reliability Engineering* (pp. 83–89).
- Kilgarriff, A. (2001). Comparing corpora. *International Journal of Corpus Linguistics*, 6(1), 97–133. doi:10.1075/ijcl.6.1.05kil
- Kilgarriff, A., & Rosenzweig, J. (2000). Framework and results for English SENSEVAL. *Computers and the Humanities*, 34(1/2), 15–48. doi:10.1023/A:1002693207386
- Kim, B.-O., & Lee, S. M. (1995). Bond rating expert system for industrial companies. *Expert Systems with Applications*, 9(1), 63–70. doi:10.1016/0957-4174(94)00049-2
- Kim, H., Golub, G. H., & Park, H. (2004). Imputation of missing values in DNA microarray gene expression data. In *Proceedings of the IEEE Computational Systems Bioinformatics Conference*, Standford, CA, USA (pp. 572–573).
- Kim, J.-D., Ohta, T., & Tsujii, J. (2008). Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*, 9(10), e25.
- Kim, K.-J. (2006). Artificial neural networks with evolutionary instance selection for financial forecasting. *Expert Systems with Applications*, 30(3), 519–526. doi:10.1016/j.eswa.2005.10.007
- Kim, S.-M., & Hovy, E. (2004). *Determining the sentiment of opinions*. Paper presented at the 20th International Conference on Computational Linguistics (COLING-04).
- Kim, S.-M., & Hovy, E. (2007). *CRYSTAL: Analyzing predictive opinions on the Web*. Paper presented at the Conference on Empirical Methods in Natural Language Processing (EMNLP-CoNLL 2007).

Compilation of References

- Kingdon, J. (1997). *Intelligent systems and financial forecasting*. London: Springer-Verlag.
- Kise, K., Sato, A., & Iwata, M. (1998). Segmentation of page images using the Area Voronoi Diagram. *Computer Vision and Image Understanding*, 70(3), 370–382. doi:10.1006/cviu.1998.0684
- Kitchenham, B., Pickard, L. M., Linkman, S., & Jones, P. W. (2003). Modeling software bidding risks. *IEEE Transactions on Software Engineering*, 29(6), 542–554. doi:10.1109/TSE.2003.1205181
- Knight, K., & Luk, S. (1994). *Building a large knowledge base for machine translation*. Paper presented at the American Association for Artificial Intelligence Conference (AAAI-94)
- Knoblock, C. (1990). Learning abstraction hierarchies for problem solving. In *Proceedings of the Seventh International Workshop on Machine Learning*.
- Knowles, J. (2006). ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multi-objective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1), 50–66. doi:10.1109/TEVC.2005.851274
- Knowles, J. D., & Corne, D. W. (2000). Approximating the nondominated front using the Pareto archived evolution strategy. *Evolutionary Computation*, 8, 149–172. doi:10.1162/106365600568167
- Kodratoff, Y., & Ganascia, J.-G. (1986). Improving the generalization step in learning. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine learning: An artificial intelligence approach: Volume II* (pp. 215–244). Los Altos, CA: Kaufmann.
- Koduru, P., Das, S., & Welch, S. M. (2007). Multi-objective and hybrid PSO using ϵ -fuzzy dominance. In D. Thierens, et al. (Eds.), *Proceedings of the ACM Genetic and Evolutionary Computing Conference*, London, UK (pp. 853–860).
- Koduru, P., Das, S., Welch, S. M., & Roe, J. (2004). Fuzzy dominance based multi-objective GA-Simplex hybrid algorithms applied to gene network models. In K. Deb, et al. (Eds.), *Proceedings of the Genetic and Evolutionary Computing Conference*, Seattle, Washington (LNCS 3102, pp. 356–367). Berlin, Germany: Springer-Verlag.
- Koduru, P., Das, S., Welch, S. M., Roe, J., & Lopez-Dee, Z. P. (2005). A co-evolutionary hybrid algorithm for multi-objective optimization of gene regulatory network models. In *Proceedings of the Genetic and Evolutionary Computing Conference*, Washington, DC (pp. 393–399).
- Koduru, P., Dong, Z., Das, S., Welch, S. M., & Roe, J. (2008). Multi-objective evolutionary-simplex hybrid approach for the optimization of differential equation models of gene networks. *IEEE Transactions on Evolutionary Computation*, 12(5), 572–590. doi:10.1109/TEVC.2008.917202
- Koduru, P., Welch, S. M., & Das, S. (2007). A particle swarm optimization approach for estimating confidence regions. In D. Thierens, et al. (Eds.), *Proceedings of the Genetic and Evolutionary Computing Conference*, London, UK (pp. 70–77).
- Kohonen, T. (1982). Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43, 59–69. doi:10.1007/BF00337288
- Kohonen, T. (1989). *Self-organization and associative memory* (3rd ed.). Berlin, Germany: Springer Verlag.
- Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, 78(9), 1464–1480. doi:10.1109/5.58325
- Kohonen, T. (1997). *Self-organizing maps*. Berlin, Germany: Springer.
- Kohonen, T. (2001). *Self-organizing maps* (3rd ed.). Berlin, Germany: Springer-Verlag.
- Kohonen, T., Kaski, S., Lagus, K., Salojarvi, J., Honkela, J., Paatero, V., & Saarela, A. (2000). Self organization of a massive document collection. *IEEE Transactions on Neural Networks*, 11(3), 574–585. doi:10.1109/72.846729
- Kokaram, A., Rea, N., Dahyot, R., Tekalp, M., Bouthemy, P., & Gros, P. (2006). Browsing sports video: Trends in sports-related indexing and retrieval work. *IEEE*

- Signal Processing Magazine*, 23(2), 47–58. doi:10.1109/MSP.2006.1621448
- Kondadadi, R., & Kozma, R. (2002). A modified fuzzy ART for soft document clustering. In . *Proceedings of International Joint Conference on Neural Networks, 2002*, 2545–2549.
- Konishi, S., Ando, T., & Imoto, S. (2004). Bayesian information criteria and smoothing parameter selection in radial basis function networks . *Biometrika*, 91(1), 27–43.
- Koonin, E., & Galperin, M. (2003). *Sequence-evolution-function: Computational approaches in comparative genomics*. Amsterdam: Kluwer Academic Publishers.
- Koren, Y., & Carmel, L. (2004). Robust linear dimensionality reduction. *IEEE Transactions on Visualization and Computer Graphics*, 10(4), 459–470. doi:10.1109/TVCG.2004.17
- Korf, R. E. (1985). Macro-operators: A weak method for learning. *Artificial Intelligence*, 26, 35–77. doi:10.1016/0004-3702(85)90012-8
- Korhonen, A., Krymolowski, Y., & Collier, N. (2008). *The Choice of Features for Classification of Verbs in Biomedical Texts*. Paper presented at the International Conference on Computational Linguistics (COLING-2008).
- Kostiainen, T., & Lampinen, J. (2002). On the generative probability density model in the self-organizing map. *Neurocomputing*, 48(1-4), 217–228. doi:10.1016/S0925-2312(01)00649-X
- Koza, J. R. (1992). *Genetic programming: On the programming of computers by means of natural selection*. Cambridge, MA: MIT Press.
- Krause, S., & Polikar, R. (2003). An ensemble of classifiers for the missing feature problem. In *Proceedings of the International Joint Conference on Neural Networks, Portland, USA* (pp. 553-558).
- Krogh, A. (1998). An introduction to Hidden Markov Models for biological sequences. In *Computational methods in molecular biology* (pp. 45-63). St. Louis, MO: Elsevier.
- Kruskal, J. B. (1964). Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis. *Psychometrika*, 29, 1–27. doi:10.1007/BF02289565
- Kruskal, J., & Liberman, M. (1983). *The symmetric time-warping problem: From continuous to discrete*. Reading, MA: Addison-Wesley.
- K-Team. (2003). *Khepera robot*. Retrieved from <http://www.k-team.com>
- Ku, C. C., & Lee, K. Y. (1995). Diagonal recurrent neural networks for dynamic system control. *IEEE Transactions on Neural Networks*, 6(1), 144–156. doi:10.1109/72.363441
- Ku, C. C., Lee, K. Y., & Edwards, R. M. (1992). Improved nuclear reactor temperature control using diagonal recurrent neural networks. *IEEE Transactions on Nuclear Science*, 39(6), 2298–2308. doi:10.1109/23.211440
- Kuan, C. M., & Liu, T. (1995). Forecasting exchange rates using feedforward and recurrent neural networks. *Journal of Applied Econometrics*, 10(4), 347–364. doi:10.1002/jae.3950100403
- Kucera, H., Francis, W., & Carroll, J. (1967). *Computational analysis of present-day American English*. Providence, RI: Brown University Press.
- Kudo, T., & Matsumoto, Y. (2003). *Fast methods for kernel-based text analysis*. Paper presented at the 41st Annual Meeting on Association for Computational Linguistics.
- Kukkonen, S. (2007). Ranking-dominance and many-objective optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2007)*, Singapore (pp. 3983-3990).
- Kung, S. Y., Mak, M. W., & Lin, S. H. (2005). *Biometric authentication: A machine learning approach*. Upper Saddle River, NJ: Prentice Hall Professional Technical Reference.
- Kurfess, F. J. (2000). Neural networks and structured knowledge: Rule extraction and applications. *Applied Intelligence*, 12(1/2), 7–13. doi:10.1023/A:1008344602888

Compilation of References

- Kwak, N., & Choi, C.-H. (2002). Input feature selection by mutual information based on parzen window. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12), 1667–1671. doi:10.1109/TPAMI.2002.1114861
- Kwok, T., & Yeung, D. (1997). Constructive algorithms for structure learning in feed forward neural networks for regression problems: A survey. *IEEE Transactions on Neural Networks*, 8(3), 630–645. doi:10.1109/72.572102
- Lachiche, N., & Flach, P. A. (2003). Improving accuracy and cost of two-class and multi-class probabilistic classifiers using ROC curves. In *Proceedings of the International Conference on Machine Learning* (pp. 416–423).
- Lacson, E., Ofsthun, N., & Lazarus, J. M. (2003). Effect of variability in anemia management on hemoglobin outcomes in ESRD. *American Journal of Kidney Diseases*, 41(1), 111–124. doi:10.1053/ajkd.2003.50030
- Lafferty, J., McCallum, A., & Pereira, F. (2001). *Conditional random fields: Probabilistic models for segmenting and labeling sequence data*. Paper presented at the Eighteenth International Conference on Machine Learning (ICML).
- Lai, C.-C., & Tsai, M.-C. (2004). *An empirical performance comparison of machine learning methods for spam e-mail categorization*. Paper presented at the Fourth International Conference on Hybrid Intelligent Systems (HIS'04).
- Lai, J. S., Luh, J. J., Lee, J. F., Chen, S. C., & Kuo, T. S. (1999, August). A DRNN based FES controller for joint position control – a computer study. In *Proceedings of the 4th Annual Conference of the International Functional Electrical Stimulation Society*, Sendai, Japan.
- Lajbcygier, P. (2004). Improving option pricing with the product constrained hybrid neural network. *IEEE Transactions on Neural Networks*, 15(2), 465–476. doi:10.1109/TNN.2004.824265
- Lakshminarayan, K., Harp, S. A., & Samad, T. (2004). Imputation of missing data in industrial databases. *Applied Intelligence*, (11): 259–275.
- Lanckriet, G., Cristianini, N., Bartlett, P., Ghaoui, L. E., & Jordan, M. (2002). Learning the kernel matrix with semi-definite programming. In *Proc. of the 19th International Conference on Machine Learning* (pp. 323–330).
- Landauer, T., & Dumais, S. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104(2), 211–240. doi:10.1037/0033-295X.104.2.211
- Landes, S., Leacock, C., & Tengi, R. (1998). Building a semantic concordance of English. In C. Fellbaum (Ed.), *WordNet: An electronic lexical database and some applications*. Cambridge, MA: MIT Press.
- Langlais, P., & Patry, A. (2007). *Translating unknown words by analogical learning*. Paper presented at the Empirical Methods in Natural Language Processing (EMNLP-CoNLL 2007).
- Langley, P., & Choi, D. (2006). A Unified cognitive architecture for physical agents. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence (AAAI'2006)*.
- Lapata, M., & Brew, C. (2004). Verb class disambiguation using informative priors. *Computational Linguistics*, 30(1), 45–73. doi:10.1162/089120104773633385
- Lapedes, A., & Farber, R. (1987). *Non-linear signal processing using neural networks: Prediction and system modeling* (Tech. Rep. LA-UR-87-2662). Los Alamos National Laboratory.
- Laven, K., Leishman, S., & Roweis, S. T. (2005). A statistical learning approach to document image analysis. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 357–361). Washington, DC: IEEE Computer Society.
- Lawrence, C., & Reilly, A. (1990). An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned byopolimer sequences. *Proteins*, 7(1), 41–51. doi:10.1002/prot.340070105
- Lazarescu, M., & Venkatesh, S. (2003, July). *Using camera motion to identify types of American football*

- plays.* Paper presented at the 2003 International Conf. on Multimedia and Expo.
- Leckie, C., & Zukerman, I. (1991). Learning search control rules for planning: An inductive approach. In *Proceedings of the Eighth International Workshop on Machine Learning* (pp. 422-426).
- Ledley, R. S., & Lusted, L. B. (1959). Reasoning foundations of medical diagnosis. *Science, 130*, 9–21. doi:10.1126/science.130.3366.9
- Lee, C. C. (1990). Fuzzy logic in control systems: Fuzzy logic controller, part I and part II. *IEEE Transactions on Systems, Man, and Cybernetics, 20*, 404–435. doi:10.1109/21.52551
- Lee, H. C., & Dagli, C. H. (1997). A parallel genetic-neuro scheduler for job-shop scheduling problems. *International Journal of Production Economics, 51*(1-2), 115–122. doi:10.1016/S0925-5273(97)00073-X
- Lee, J. (1999). A practical radial basis function equalizer. *IEEE Transactions on Neural Networks, 10*, 450–455.
- Lee, M., Wang, W., & Yu, H. (2006). Exploring supervised and unsupervised methods to detect topics in biomedical text. *BMC Bioinformatics, 7*(140).
- Lee, S.-W., & Li, S. Z. (2007). *Advances in biometrics* (LNCS 4642). Berlin, Germany: Springer Verlag.
- Lee, Y., & Ng, H. (2002). *An empirical evaluation of knowledge sources and learning algorithms for word sense disambiguation*. Paper presented at the Conference on Empirical Methods in Natural Language Processing (EMNLP-2002).
- Leech, G., Garside, R., & Bryant, M. (1994). *CLAWS4: The tagging of the British National Corpus*. Paper presented at the 15th International Conference on Computational Linguistics (COLING 94).
- Leemans, V., & Destain, M. F. (2004). A real-time grading method of apples based on features extracted from defects. *Journal of Food Engineering, 61*(1), 83–89. doi:10.1016/S0260-8774(03)00189-4
- Leemans, V., Magein, H., & Destain, M. F. (1999). Defect segmentation on 'Jonagold' apples using colour vision and a Bayesian classification method. *Computers and Electronics in Agriculture, 23*, 43–53. doi:10.1016/S0168-1699(99)00006-X
- Leiden, J. (2008). *Toponym resolution in text: Annotation, evaluation and applications of spatial grounding of place names*. Retrieved from <http://www.dissertation.com>
- Leighton, R. R., & Conrath, B. C. (1991). The autoregressive backpropagation algorithm. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN, ii*, 369–377. doi:10.1109/IJCNN.1991.155362
- Leistritz, L., Kochs, E., Galicki, M., & Witte, H. (2002). Prediction of movement following noxious stimulation during 1 minimum alveolar anesthetic concentration isoflurane/nitrous oxide anesthesia by means of middle latency auditory evoked responses. *Clinical Neurophysiology, 113*(6), 930–935. doi:10.1016/S1388-2457(02)00064-0
- Lemmon, D., Shiang, T. Y., Hashmi, A., Ulbrecht, J. S., & Cavanagh, P. R. (1997). The effect of insoles in therapeutic footwear - a finite element approach. *Journal of Biomechanics, 30*, 615–620. doi:10.1016/S0021-9290(97)00006-7
- Lenstra, J. (1974). Clustering a data array and the traveling salesman problem. *Operations Research, 22*(2), 413–414. doi:10.1287/opre.22.2.413
- Lesk, A. M. (2002). *Introduction to bioinformatics*. Oxford, UK: Oxford University Press.
- Leung, Y. F., & Cavalieri, D. (2003). Fundamentals of cDNA microarray data analysis. *Trends in Genetics, 19*(11), 649–659. doi:10.1016/j.tig.2003.09.015
- Levin, B. (1993). *English verb classes and alterations: A preliminary investigation*. Chicago, IL: The Chicago University Press.
- Li, N., & Tompa, M. (2006). Analysis of computational approaches for motif discovery. *Algorithms for Molecular Biology; AMB, 1*(8). doi:10.1007/978-1-59745-000-3
- Li, Y., & Shawe-Taylor, J. (2006). Using KCCA for Japanese---English cross-language information retrieval and document classification. *Journal of Intelligent In-*

Compilation of References

- formation Systems*, 27(2), 117–133. doi:10.1007/s10844-006-1627-y
- Li, Y., Bontcheva, K., & Cunningham, H. (2007). *Hierarchical, perceptron-like learning for ontology based information extraction*. Paper presented at the Proceedings of the 16th International World Wide Web Conference (WWW2007).
- Liano, K. (1996). Robust error measure for supervised neural network learning with outliers. *IEEE Transactions on Neural Networks*, 7(1), 246–250. doi:10.1109/72.478411
- Liao, T. W. (2005). Clustering of time series data - a survey. *Pattern Recognition*, 38(11), 1857–1874. doi:10.1016/j.patcog.2005.01.025
- Lin, C.-F., & Wang, S.-D. (2002). Fuzzy support vector machines. *IEEE Transactions on Neural Networks*, 13(2).
- Lin, D. (1998). An information-theoretic definition of similarity. In *Proceedings of the 15th International Conference on Machine Learning* (pp. 296-304). San Francisco, CA: Morgan Kaufmann.
- Lin, D. (1998). *Dependency-based evaluation of MINIPAR*. Paper presented at the Workshop on the Evaluation of Parsing Systems.
- Lin, D., & Pantel, P. (2001). Discovery of inference rules for question answering. *Natural Language Engineering*, 7(4), 343–360. doi:10.1017/S1351324901002765
- Lin, D., & Pantel, P. (2002). *Concept discovery from text*. Paper presented at the Conference on Computational Linguistics (COLING-02).
- Lin, G. F., & Chen, L. H. (2004). A non-linear rainfall-runoff model using radial basis function network. *Journal of Hydrology (Amsterdam)*, 289, 1–8.
- Lin, S., & Kernighan, B. (1973). An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21, 498–516. doi:10.1287/opre.21.2.498
- Lin, T., Horne, B. G., & Giles, C. L. (1998). How embedded memory in recurrent neural network architectures helps learning long-term temporal dependencies. *Neural Networks*, 11, 861–868. doi:10.1016/S0893-6080(98)00018-5
- Linares-Barranco, B., Serrano-Gotarredona, M., & Andreou, A. (1998). *Adaptive resonance theory micro-chips: Circuit design techniques*. Norwell, MA: Kluwer Academic Publisher.
- Lisboa, P. J. G. (2002). A review of evidence of health benefit from artificial neural networks in medical intervention. *Neural Networks*, 15(1), 11–39. doi:10.1016/S0893-6080(01)00111-3
- Little, R. J. A., & Rubin, D. B. (1987). *Statistical analysis with missing data*. New York: John Wiley.
- Little, R. J. A., & Rubin, D. B. (2002). *Statistical analysis with missing data* (2nd ed.). New York: John Wiley & Sons.
- Liu, B. (2006). *Web data mining - exploring hyperlinks, contents and usage data*. Berlin, Germany: Springer.
- Liu, P., El-Darzi, E., Lei, L., Vasilakis, C., Chountas, P., & Huang, W. (2005). An analysis of missing data treatment methods and their application to health care dataset. In X. Li, et al. (Eds.), *Advance data mining applications* (LNCS 3584, pp. 583-590). Berlin, Germany: Springer.
- Liu, S., & Du, Y. (2005). Sonar array servo system based on diagonal recurrent neural network. In *Proceedings of the IEEE International Conference on Mechatronics & Automatics*, Niagara Falls, Ontario, Canada (pp. 1912-1917).
- Liu, S., Wang, Y., & Zhu, Q. M. (2008). Development of a new EDRNN procedure in control of human arm trajectories. *Neurocomputing*, 72, 490–499. doi:10.1016/j.neucom.2007.12.012
- Liu, Z.-Y., Chiu, K.-C., & Xu, L. (2003). Improved system for object detection and star/galaxy classification via local subspace analysis. *Neural Networks*, 16, 437–451. doi:10.1016/S0893-6080(03)00015-7
- Ljung, L. (1999). *System identification. Theory for the user*. Upper Saddle River, NJ: Prentice Hall.

- Lloyd, J. W. (1987). *Foundations of logic programming* (2nd ed.). New York: Springer-Verlag.
- Lloyd, S. (1957). *Least square quantization in PCM's* (Bell Telephone Laboratories Paper).
- Loève, M. M. (1955). *Probability theory*. Princeton, NJ: VanNostrand.
- Lonardi, S. (2002). Pattern discovery in biosequences - tutorial. In *Proceedings of the 10th International Conference on Intelligent Systems for Molecular Biology*.
- Lopez, A. (2008). Statistical machine translation. *ACM Computing Surveys*, 40(3). doi:10.1145/1380584.1380586
- Lotem, A., & Nau, D. (2000). New advances in GraphHTN: Identifying independent subproblems in large HTN domains. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems* (pp. 206-215).
- Löwe, M. (1993). Algebraic approach to single-pushout graph transformation. *Theoretical Computer Science*, 109, 181–224. doi:10.1016/0304-3975(93)90068-5
- Lowrie, E. G., Ling, J., Lew, N. L., & Yiu, Y. (1994). The relative contribution of measured variables to death risk among hemodialysis patients. In E. A. Friedman (Ed.), *Death on hemodialysis: preventable or inevitable?* (pp. 121-141). Boston, MA: Kluwer Academic Publishers.
- Lu, Y., & Tan, C. L. (2005). Constructing Area Voronoi Diagram in document images. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 342-346). Washington, DC: IEEE Computer Society.
- Luenberger, D. (1984). *Linear and nonlinear programming* (2nd ed.). Reading, MA: Addison Wesley.
- Luh, G. C., & Chen, W. (2005). Immune model-based fault diagnosis. *Mathematics and Computers in Simulation*, 67(6), 515–539. doi:10.1016/j.matcom.2004.07.004
- Lumley, J. L. (1967). The structure of inhomogeneous turbulent flows. In A. M. Yaglom & V. I. Tatarski (Eds.), *Atmospheric turbulence and radio propagation* (pp. 166-178). Moscow, Russia: Nauka.
- Luxhoj, J. T., Riis, J. O., & Stensballe, B. (1996). A hybrid econometric-neural network modeling approach for sales forecasting. *International Journal of Production Economics*, 43(2-3), 175–192. doi:10.1016/0925-5273(96)00039-4
- Ma, S., Ji, C., & Farmer, J. (1997). An Efficient EM-based Training Algorithm for Feedforward Neural Networks . *Neural Networks*, 10(2), 243–256.
- Macciotta, N. P. P., Cappio-Borlino, A., & Pulina, G. (2000). Time series autoregressive integrated moving average modelling of test-day milk of dairy ewes. *Journal of Dairy Science*, 83, 1094–1103.
- MacKay, D. (2003). *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press.
- MacKay, D. J. C. (1992). A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3), 448–472. doi:10.1162/neco.1992.4.3.448
- MacKay, D. J. C. (1997). *Ensemble learning for Hidden Markov Models* (Tech. Rep.). Cavendish Laboratory, University of Cambridge, U.K.
- Mackey, D. (1992). A practical Bayesian framework for backpropagation . *Neural Computation*, 4, 448–472.
- MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium*, 1 (pp. 281-297).
- Mai-Duy, N., & Tran-Cong, T. (2001). Numerical solution of differential equations using multiquadric radial basis function networks . *Neural Networks*, 14(2), 185–199.
- Majumdar, S., & Jayas, D. S. (2000). Classification of cereal grains using machine vision: IV. Combined morphology, color, and texture models. *Transactions of the American Society of Agricultural Engineers*, 43(6), 1689–1694.
- Mak, M. W. (1995). A learning algorithm for recurrent radial basis functions networks. *Neural Processing Letters*, 2(1), 27–31. doi:10.1007/BF02312380
- Mak, M. W., & Kung, S. Y. (2000). Estimation of elliptical basis function parameters by the EM algorithm with appli-

Compilation of References

- cation to speaker verification. *IEEE Transactions on Neural Networks*, 11(4), 961–969. doi:10.1109/72.857775
- Makridakis, S. G., Wheelwright, S. C., & Hyndman, R. J. (1997). *Forecasting: Methods and applications*. New York: John Wiley & Sons.
- Makridakis, S., Anderson, A., Carbone, R., Fildes, R., Hibdon, M., & Lewandowski, R. (1982). The accuracy of extrapolation (time series) methods: Results of a forecasting competition. *Journal of Forecasting*, 1, 111–153. doi:10.1002/for.3980010202
- Mandic, D. P., & Chambers, J. A. (2001). *Recurrent neural networks for prediction*. Chichester, UK: John Wiley & Sons Ltd.
- Manning, C., & Schütze, H. (2003). *Foundations of statistical natural language processing* (6th ed.). Cambridge, MA: The MIT Press.
- Mansfield, A. J., & Wayman, J. L. (2002). *Best practices in testing and reporting performance of biometric devices*. Centre for Mathematics and Scientific Computing, National Physical Laboratory.
- Marchant, J. A., & Onyango, C. M. (2003). Comparison of a Bayesian classifier with a multilayer feed-forward neural network using the example of plant/weed/soil discrimination. *Computers and Electronics in Agriculture*, 39, 3–22. doi:10.1016/S0168-1699(02)00223-5
- Marco, C., Dorigo, M., & Borghi, G. (1996). Behavior analysis and training – a methodology for behavior engineering. *IEEE Transactions on Systems, Man, and Cybernetics*, 26(6), 365–380.
- Marcus, M., Santorini, B., & Marcinkiewicz, M. (1993). Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2), 313–330.
- Marinai, S. (2008). Self-organizing maps for clustering in document image analysis. In S. Marinai & H. Fujisawa (Eds.), *Machine learning in document analysis and recognition* (pp. 381-408). Berlin, Germany: Springer.
- Marinai, S., Gori, M., & Soda, G. (2005). Artificial neural networks for document analysis and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1), 23–35. doi:10.1109/TPAMI.2005.4
- Marinai, S., Marino, E., & Soda, G. (2006). Tree clustering for layout based document image retrieval. In *Proceedings of the 2nd International Workshop on Document Image Analysis for Libraries* (pp. 243-253). Washington, DC: IEEE Computer Society.
- Markey, M. K., & Patel, A. (2004). Impact of missing data in training artificial neural networks for computer-aided diagnosis. In *Proceedings of the International Conference on Machine Learning Applications*, Louisville, KY (pp. 351-354).
- Martin, M., & Geffner, H. (2000). Learning generalized policies in planning using concept languages. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning Systems (AIPS00)*.
- Martinetz, T., & Schulten, K. (1994). Topology representing networks. *Neural Networks*, 7(3), 507–522. doi:10.1016/0893-6080(94)90109-0
- Martinetz, T., Berkovich, S., & Schulten, K. (1993). “Neural-Gas” network for vector quantization and its application to time-series prediction. *IEEE Transactions on Neural Networks*, 4(4), 558–569. doi:10.1109/72.238311
- Martínez-de-Pisón, F. J. (2003). *Optimización mediante técnicas de minería de datos del ciclo de recocido de una línea de galvanizado*. Unpublished doctoral dissertation, University of La Rioja, Spain.
- Martínez-de-Pisón, F. J., Alba, F., Castejón, M., & González, J. A. (2006). Improvement and optimisation of a hot dip galvanizing line using neural networks and genetic algorithms. *Ironmaking & Steelmaking*, 33(4), 344–352. doi:10.1179/174328106X101565
- Martínez-de-Pisón, F. J., Ordieres, J., Pernía, A., Alba, F., & Torre, V. (2007). Reducción de problemas de adherencia en procesos de galvanizado mediante técnicas de minería de datos. *Revista de Metalurgia*, 43(5), 325–336. doi:10.3989/revmetalm.2007.v43.i5.77
- Martín-Guerrero, J. D., Camps-Valls, G., Soria-Olivas, E., Serrano-López, A. J., Pérez-Ruixo, J. J., & Jiménez-

- Torres, N. V. (2003a). Dosage individualization of erythropoietin using a profile-dependent support vector regression. *IEEE Transactions on Bio-Medical Engineering*, 50(10), 1136–1142. doi:10.1109/TBME.2003.816084
- Martín-Guerrero, J. D., Soria-Olivas, E., Camps-Valls, G., Serrano-López, A. J., Pérez-Ruixo, J. J., & Jiménez-Torres, N. V. (2003b). Use of neural networks for dosage individualisation of erythropoietin in patients with secondary anemia to chronic renal failure. *Computers in Biology and Medicine*, 33(4), 361–373. doi:10.1016/S0010-4825(02)00065-3
- Martin-Guerrero, J. D., Soria-Olivas, E., Chorro-Mari, V., Climente-Martí, M., & Jimenez-Torres, N. V. (2006). Reinforcement learning for anemia management in hemodialysis patients treated with erythropoietic stimulating factors. In *Proceedings of the 17th European Conference on Artificial Intelligence ECAI 2006*, Riva del Garda, Italy (pp. 19-24).
- Massey, L. (2003). On the quality of ART1 text clustering. *Neural Networks*, 16, 771–778. doi:10.1016/S0893-6080(03)00088-1
- Matsatsinis, N. F., Doumpos, M., & Zopounidis, C. (1997). Knowledge acquisition and representation for expert systems in the field of financial analysis. *Expert Systems with Applications*, 12(2), 247–262. doi:10.1016/S0957-4174(96)00098-X
- Maurer, D. E., & Baker, J. P. (2007). Fusing multi-modal biometrics with quality estimates via a Bayesian belief network. *Pattern Recognition*, 41(3), 821–832. doi:10.1016/j.patcog.2007.08.008
- Mawudeku, A., & Blench, M. (2005). *Global public health intelligence network (GPHIN)*, invited talk. Paper presented at the The Tenth Machine Translation Summit.
- May-Plumlee, T., & Little, T. J. (2001). consumer purchase data as a strategic product development tool. *Journal of Textile and Apparel, Technology and Management*, 1(3), 1–10.
- Mcallester, D., & Givan, R. (1993). Taxonomic syntax for first order inference. *Journal of the ACM*, 40, 289–300.
- McCormick, W., Schweitzer, P., & White, T. (1972). Problem decomposition and data reorganization by a clustering technique. *Operations Research*, 20, 993–1009. doi:10.1287/opre.20.5.993
- McCreight, E. (1976). A space economical space tree construction algorithm. *Journal of the ACM*, 23(2), 262–272. doi:10.1145/321941.321946
- McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5, 115–133. doi:10.1007/BF02478259
- McDonnell, J. R., & Waagen, D. (1994). Evolving recurrent perceptrons for time-series modeling. *IEEE Transactions on Neural Networks*, 5(1), 24–38. doi:10.1109/72.265958
- McGann, C., Py, F., Rajan, K., Ryan, H., & Henthorn, R. (2008). Adaptative control for autonomous underwater vehicles. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*.
- McLachlan, G. J., & Krishnan, T. (1997). *The EM algorithm and extensions*. New York: Wiley.
- McMullen, P. R. (2001). A Kohonen self-organizing map approach to addressing a multiple objective, mixed-model JIT sequencing problem. *International Journal of Production Economics*, 72(1), 59–71. doi:10.1016/S0925-5273(00)00091-8
- Mei, T., Ma, Y., Zhou, H., Ma, W., & Zhang, H. (2005). *Sports video mining with Mosaic*. Paper presented at the 11th IEEE International Multimedia Modeling Conference.
- Mel, B. W., & Omohundro, S. M. (1991). How receptive field parameters affect neural learning. In Lippmann, Moody, & Touretzky (eds), *Advances in Neural Information Processing System 3*, Morgan Kaufmann Pub., 757-763.
- Mena, Y., Castel, J. M., Caravaca, F. P., Guzmán, J. L., & González, P. (2005). Situación actual, evolución y diagnóstico de los sistemas semiextensivos de producción caprina en Andalucía centro-occidental. In *Junta*

Compilation of References

- de Andalucía. Consejería de agricultura y pesca (pp. 222).
- Mercer, J. (1909). Functions of positive and negative type, and their connection with the theory of integral equations. *Transactions of the London Philosophical Society (A)*, 209, 415–446. doi:10.1098/rsta.1909.0016
- Meyer, D., Leisch, F., & Hornik, K. (2003). The support vector machine under test. *Neurocomputing*, 55, 169–186. doi:10.1016/S0925-2312(03)00431-4
- Michalski, R. S. (1994). Inferential theory of learning. Developing foundations for multistrategy learning. In R. S. Michalski & G. Tecuci (Eds.), *Machine learning: a multistrategy approach, volume IV* (pp. 3-61). San Mateo, CA: Morgan Kaufmann.
- Michel, O. (1998). Webots: Symbiosis between virtual and real mobile robots. In *Proceedings of the First International Conference on Virtual Worlds (VW'98)*, Paris, France (pp. 254-253). Berlin, Germany: Springer Verlag.
- Michel, O., & Cyberbotics, Ltd. (2004). WebotsTM: Professional mobile robot simulation. *International Journal of Advanced Robotic Systems*, 1(1), 39–42.
- Mihalcea, R., & Csomai, A. (2005). *SenseLearner: Word sense disambiguation for all words in unrestricted text*. Paper presented at the Annual Meeting of the Association for Computational Linguistics (ACL-2005).
- Mika, S., Rätsch, G., Weston, J., Schölkopf, B., & Müller, K. R. (1999). Fisher discriminant analysis with kernels. In Y. H. Hu, J. Larsen, E. Wilson, & S. Douglas (Eds.), *Advances in neural networks for signal processing* (pp. 41-48). San Mateo, CA: Morgan Kaufmann.
- Milán, M. J., Arnalte, A., & Caja, G. (2003). Economic profitability and typology of Rиполеса breed sheep faros in Spain. *Small Ruminant Research*, 49, 97–105. doi:10.1016/S0921-4488(03)00058-0
- Miller, G. F., Todd, P. M., & Hegde, S. U. (1989). Designing neural networks using genetic algorithms. In J. D. Schaffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms* (pp. 370-384). San Francisco: Morgan Kaufman.
- Minkov, E., Wang, R., & Wang, W. (2005). *Extracting personal names from email: Applying named entity recognition to informal text*. Paper presented at the Empirical Methods in Natural Language Processing.
- Minton, S. (1988). *Learning effective search control knowledge: an explanation based approach*. Amsterdam: Kluwer Academic Publisher.
- Mirkin, B. (2005). *Clustering for data mining: A data recovery approach*. Boca Raton, FL: Chapman and Hall.
- Mitchell, M. (1996). *An introduction to genetic algorithms*. Cambridge, MA: MIT Press.
- Mitchell, R. S., Sherlock, R. A., & Smith, L. A. (1996). An investigation into the use of machine learning for determining oestrus in cows. *Computers and Electronics in Agriculture*, 15(3), 195–213. doi:10.1016/0168-1699(96)00016-6
- Mitchell, T. (1997). *Machine learning*. New York: McGraw-Hill.
- Mitchell, T. M. (1996). *An introduction to genetic algorithms*. Cambridge, MA: MIT Press.
- Mitchell, T. M. (2006). *The discipline of machine learning*. Pittsburgh, PA: Carnegie Mellon University.
- Mitchell, T., Utgoff, T., & Banerjiartin, R. (1982). Learning problem solving heuristics by experimentation. In *Machine learning: An artificial intelligence approach*.
- Mitkov, R. (Ed.). (2003). *The Oxford handbook of computational linguistics*. Oxford, UK: Oxford University Press.
- Mitra, S. (1994). Fuzzy MLP based expert system for medical diagnosis. *Fuzzy Sets and Systems*, 65, 285–296. doi:10.1016/0165-0114(94)90025-6
- Mitra, S., & Hayashi, Y. (2000). Neuro-fuzzy rule generation: Survey in soft computing framework. *IEEE Transactions on Neural Networks*, 11(3), 748–768. doi:10.1109/72.846746

- Miyao, Y., Sætre, R., Sagae, K., Matsuzaki, T., & Tsujii, J. (2008). *Task-oriented evaluation of syntactic parsers and their representations*. Paper presented at the Association for Computational Linguistics.
- Miyazaki, Y., Terakado, M., Ozaki, K., & Nozaki, N. (1994). Robust regression for developing software estimation models. *Journal of Systems and Software*, 27(1), 3–16. doi:10.1016/0164-1212(94)90110-4
- Mladenic, D. (2002). *Automatic word lemmatization*. Paper presented at the 5th International Multi-Conference: Information Society (IS 2002).
- Mohammed, H. S., Stepenosky, N., & Polikar, R. (2006). An ensemble technique to handle missing data from sensors. In *Proceedings of the IEEE Sensor Applications Symposium*, Houston, TX, USA (pp. 101-105).
- Montgomery, D. C., Peck, E. A., & Vining, G. G. (2001). *Introduction to Linear regression analysis*. New York: John Wiley & Sons, Inc.
- Moody, J., & Darken, C. J. (1989). Fast learning in networks of locally tuned processing units. *Neural Computation*, 1(2), 281–294. doi:10.1162/neco.1989.1.2.281
- Moore, B. (1989). ART1 and pattern clustering. In *Proceedings of the 1988 Connectionist Models Summer School* (pp. 174-185).
- Mougiakakou, S. G., & Nikita, K. S. (2000). A neural network approach for insulin regime and dose adjustment in type 1 diabetes. *Diabetes Technology & Therapeutics*, 2(3), 381–389. doi:10.1089/15209150050194251
- Mozer, M. C. (1989). A focused backpropagation algorithm for temporal pattern recognition. *Complex Systems*, 3, 349–381.
- Mozer, M. C. (1993). Neural net architectures for temporal sequence processing. In A. Weigend & N. Gershenfeld (Eds.), *Predicting the future and understanding the past*. Redwood City, CA, USA: Addison Wesley.
- Muggleton, S. (1995). Inverse entailment and prolog. *New Generation Computing*, 13, 245–286. doi:10.1007/BF03037227
- Muggleton, S., & Feng, C. (1990). Efficient induction of logic programs. In *Proceedings of the 1st Conference on Algorithmic Learning Theory* (pp. 368-381).
- Mulder, S., & Wunsch, D. (2003). Million city traveling salesman problem solution by divide and conquer clustering with adaptive resonance neural networks. *Neural Networks*, 16, 827–832. doi:10.1016/S0893-6080(03)00130-8
- Mulier, F., & Cherkassky, V. (1995). Self-organization as an iterative kernel smoothing process. *Neural Computation*, 7, 1165–1177. doi:10.1162/neco.1995.7.6.1165
- Muñoz-Avila, H., Aha, D., Breslow, D., & Nau, D. (1999). HICAP: An interactive case based planning architecture and its application to non-combatant evacuation operations. In *Proceedings of the Ninth Conference on Innovative Applications of Artificial Intelligence*.
- Murphy, A. H. (1972). Scalar and vector partitions of the probability score: Part ii. N-state situation. *Journal of Applied Meteorology*, 11, 1182–1192. doi:10.1175/1520-0450(1972)011<0273:SAVPOT>2.0.CO;2
- Murphy, K. (2002). *Dynamic Bayesian networks: Representation, inference and learning*. UC Berkeley.
- Nadler, B., Lafon, S., Coifman, R., & Kevrekidis, I. G. (2008). Diffusion maps - a probabilistic interpretation for spectral embedding and clustering algorithms. In A. Gorban, B. Kegl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 238-260). Berlin, Germany: Springer.
- Nagl, M. (1976). Formal languages of labelled graphs. *Computing*, 16, 113–137. doi:10.1007/BF02241984
- Nagy, G. (2000). Twenty years of document image analysis in PAMI. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), 38–62. doi:10.1109/34.824820
- Nagy, G., Seth, S., & Viswanathan, M. (1992). A prototype document image analysis system for technical journals. *Computer*, 25(7), 10–22. doi:10.1109/2.144436
- Nallapati, R. (2004). *Discriminative models for information retrieval*. Paper presented at the 27th annual international ACM SIGIR conference on Research and

Compilation of References

- development in information retrieval, Sheffield, United Kindom.
- Nandakumar, K., Chen, Y., Dass, S. C., & Jain, A. K. (2008). Likelihood ratio-based biometric score fusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2), 342–347. doi:10.1109/TPAMI.2007.70796
- Nastase, V., & Strube, M. (2008). *Decoding Wikipedia categories for knowledge acquisition*. Paper presented at the Twenty-Third AAAI Conference on Artificial Intelligence (AAAI 2008).
- National Institutes of Health. (2007). *The human genome project*. Retrieved from <http://www.genome.gov/>
- Natschläger, T., Maass, W., & Zador, A. (2001). Efficient temporal processing with biologically realistic dynamic synapses. *Network (Bristol, England)*, 12(1), 75–87. doi:10.1080/713663153
- Natschläger, T., Maass, W., Sontag, E. D., & Zador, A. (2000, November). Processing of time series by neural circuits with biologically realistic synaptic dynamics. In *Proceedings of the Neural Information Processing Systems, NIPS 2000*, Denver, Colorado, USA (pp. 145–151).
- Natsev, A., Naphade, M. R., & Smith, J. R. (2004). *Semantic representation, search and mining of multimedia content*. Paper presented at the ACM SIGKDD.
- Nau, D., Au, T., Ilghami, O., Kuter, U., Murdock, W., Wu, D., & Yaman, F. (2003). Shop: An HTN planning system. *Journal of Artificial Intelligence Research*, 20, 379–404.
- Nauck, D., & Kruse, R. (1999). Learning in neuro-fuzzy systems with symbolic attributes and missing values. In *Proceedings of the International Conference on Neural Information Processing*, Perth, WA, Australia (pp. 142–147).
- Navarro, M. J. (2005). *Caracterización socio-económica de los sistemas de producción de caprino de la comunidad autónoma de Murcia*. Unpublished doctoral dissertation, Univ Miguel Hernández, Elche, Spain.
- Navarro, M. J., & Fernández, C. (2006). Introduction to the situation of the goat sector in Murcia Region. In *Options Méditerranées. Serie A* 70 (pp. 157–163).
- Neamatullah, I., Douglass, M., Lehman, L., Reisner, A., Villarroel, M., Long, W., Szolovits, P., Moody, G., Mark, R., & Clifford, G. (2008). Automated de-identification of free-text medical records. *Medical Informatics and Decision Making*, 8(32).
- Neath, A. A., & Cavanaugh, J. E. (1997). Regression and Time Series model selection using variants of the Schwarz information criterion . *Communications in Statistics A*, 26, 559–580.
- Nebel, B., Dimopoulos, Y., & Koehler, J. (1997). Ignoring irrelevant facts and operators in plan generation. In *Proceedings of the European conference on Planning*.
- Nebot, A., Cellier, F. E., & Linkens, D. A. (1996). Synthesis of an anaesthetic agent administration system using fuzzy inductive reasoning. *Artificial Intelligence in Medicine*, 8(2), 147–166. doi:10.1016/0933-3657(95)00030-5
- Nefian, A. V., Liang, L., Pi, X., Liu, X., & Murphy, K. (2002). Dynamic Bayesian networks for audio-visual speech recognition. *EURASIP Journal on Applied Signal Processing*, (11), 1–15.
- Nehmzow, U. (1992). *Experiments in competence acquisition for autonomous mobile robots*. Unpublished doctoral dissertation, Dept of Artificial Intelligence, University of Edinburgh.
- Nelson, A. L., & Grant, E. (2006). Developmental analysis in evolutionary robotics. In *Proceedings of the Adaptive and Learning System, IEEE Mountain Workshop on IEEE CNF* (pp. 201–206).
- Neto, J. C., Meyer, G. E., Jones, D. D., & Samal, A. K. (2006). Plant species identification using Elliptic Fourier leaf shape analysis. *Computers and Electronics in Agriculture*, 50, 121–134. doi:10.1016/j.compag.2005.09.004
- Neuwald, A., & Green, P. (1994). Detecting patterns in protein sequences. *Journal of Molecular Biology*, 239, 698–712. doi:10.1006/jmbi.1994.1407

- Newham, E. (1995). *The biometrics report*. SJB Services.
- Newton, M., Levine, J., Fox, M., & Long, D. (2007). Learning macro-actions for arbitrary planners and domains. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- Nguyen, L. N., & Scherer, W. T. (2003). *Imputation techniques to account for missing data in support of intelligent transportation systems applications* (Tech. Rep.). University of Virginia, USA.
- Nienhuys-Cheng, S. (1998). Distances and limits on Herbrand interpretations. In D. Page (Ed.), *Proceedings of the ILP-98*. Berlin, Germany: Springer.
- Nikunj, C., Ravi, V., & Karthik, C. (in press). Differential evolution trained wavelet neural networks: Application to bankruptcy prediction in banks. *Expert Systems with Applications*. doi:doi:10.1016/j.eswa.2008.09.019
- Nilsson, N. J. (1984). *Shakey the robot* (Tech. Rep. 323). AI Center, SRI International, Menlo Park, CA.
- Nio, F., Gomez, D., Wong, D., & Vejar, R. (2003). A novel immune anomaly detection technique based on negative selection. In *Artificial immune systems* (LNCS 2723, p. 204). Berlin, Germany: Springer.
- NOAA Satellite & Information Service. (2002). *Search, rescue and tracking*. Retrieved from <http://www.sarsat.noaa.gov/>
- Nolfi, S., & Floreano, D. (2000). *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. Cambridge, MA: The MIT Press.
- Noordam, J. C., Otten, G. W., Timmermans, A. J. M., & Zwol, B. V. (2000). High speed potato grading and quality inspection based on a color vision system. In K. W. Tobin Jr. (Ed.), *Machine vision applications in industrial inspection VIII, Proceedings of SPIE* (Vol. 3966) (pp. 206-220).
- Nordbotten, S. (1996). Neural network imputation applied to the Norwegian 1990 population census data. *Journal of Official Statistics*, 12, 385–401.
- Nowlan, S. J. (1990). Max likelihood competition in RBF networks, TR. CRG-Tr-90-2, U. of Toronto, Dept. of Computer Science.
- Nunez, H., Angulo, C., & Catata, A. (2002). Rule extraction from support vector machines. In *Proceedings of the European Symposium on Artificial Neural Networks* (pp. 107-112).
- Nunn, I., & White, T. (2005). The application of antigenic search techniques to time series forecasting. In *Proceedings of the Genetic and Evolutionary Computation Conference*, Washington, DC (pp. 353-360).
- O'Brien, D. B., Gupta, M. R., & Gray, R. M. (2008). Cost-sensitive multi-class classification from probability estimates. In *Proceedings of the 25th International Conference on Machine Learning* (pp. 712-719).
- O'Gorman, L. (1993). The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11), 1162–1173. doi:10.1109/34.244677
- Och, F. (2005). *Statistical machine translation: Foundations and recent advances*. Paper presented at the Machine Translation Summit.
- Oh, K., Kim, T. Y., Min, S.-H., & Lee, H. Y. (2006). Portfolio algorithm based on portfolio beta using genetic algorithm. *Expert Systems with Applications*, 30(3), 527–534. doi:10.1016/j.eswa.2005.10.010
- Ohlson, J. A. (1980). Financial ratios and the probabilistic prediction of bankruptcy. *Journal of Accounting Research*, 18, 109–131. doi:10.2307/2490395
- Okamura, N. K., Delwiche, M. J., & Thompson, J. F. (1993). Raisin grading by machine vision. *Transactions of the American Society of Agricultural Engineers*, 36(2), 485–492.
- Olier, I., & Vellido, A. (2008). Advances in clustering and visualization of time series using GTM through time. *Neural Networks*, 21(7), 904–913. doi:10.1016/j.neunet.2008.05.013
- Olier, I., & Vellido, A. (2008). Variational Bayesian generative topographic mapping. *Journal of Mathematical*

Compilation of References

- Modelling and Algorithms*, 7(4), 371-387. doi:10.1007/s10852-008-9088-7
- Olmeda, I., & Fernandez, E. (1997). Hybrid classifiers for financial multicriteria decision making: the case of bankruptcy prediction. *Computational Economics*, 10, 317–335. doi:10.1023/A:1008668718837
- On, C. K., Teo, J., & Saudi, A. (2008). Multi-objective artificial evolution of RF-localization behavior and neural structures in mobile robots. In *Proceedings of the IEEE World Congress on Computational Intelligence (WCCI-CEC 2008)*, Hong Kong.
- Onwuanyi, O. N. (2000). Calcaneal spurs and plantar heel pad pain. *The Foot*, 10, 182–185. doi:10.1054/foot.2000.0633
- Onyango, C. M., Marchant, J. A., & Zwiggelaar, R. (1997). Modeling uncertainty in agriculture image analysis. *Computers and Electronics in Agriculture*, 17, 295–305. doi:10.1016/S0168-1699(97)01322-7
- Ordieres-Meré, J. B., González-Marcos, A., González, J. A., & Lobato-Rubio, V. (2004). Estimation of mechanical properties of steel strips in hot dip galvanizing lines. *Ironmaking & Steelmaking*, 31(1), 43–50. doi:10.1179/030192304225012060
- Orr, G. B., & Müller, K. R. (1998). *Neural networks: Tricks of the trade*. Berlin, Germany: Springer-Verlag.
- Ostrovsky, R., Rabani, Y., Schulman, L. J., & Swamy, C. (2006). The effectiveness of Lloyd-type methods for the k-means problem. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science* (pp. 165-176). Washington, DC: IEEE Computer Society.
- Osuna, E. E., Freund, R., & Girosi, F. (1997). *Support vector machines: Training and applications* (Tech. Rep. AIM-1602).
- Osuna, E., Freund, R., & Girosi, F. (1997). Training support vector machines: An application to face detection. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 130-136).
- Padberg, F. (2002). Empirical interval estimates for the defect content after an inspection. In *Proceedings of the 22nd International Conference on Software Engineering* (pp. 58-68).
- Padberg, F. (2003). Maximum likelihood estimates for the hypergeometric software reliability model. *International Journal of Reliability Quality and Safety Engineering*, 10(1), 41–53. doi:10.1142/S0218539303000981
- Padberg, F., Ragg, T., & Schoknecht, R. (2004). Using machine learning for estimating the defect content after an inspection. *IEEE Transactions on Software Engineering*, 30(1), 17–28. doi:10.1109/TSE.2004.1265733
- Pal, N. (1999). Soft computing for feature analysis. *Fuzzy Sets and Systems*, 103, 201–202. doi:10.1016/S0165-0114(98)00222-X
- Pal, N., Bezdek, J., & Tsao, E. (1993). Generalized clustering networks and Kohonen's self-organizing scheme. *IEEE Transactions on Neural Networks*, 4(4), 549–557. doi:10.1109/72.238310
- Pang, B., Lee, L., & Vaithyanathan, S. (2002). *Thumbs up? Sentiment classification using machine learning techniques*. Paper presented at the Empirical Methods of Natural Language Processing (EMNLP'02).
- Panuccio, A., Bicego, M., & Murino, V. (2002). A hidden Markov model-based approach to sequential data clustering. In *Structural, syntactic and statistical pattern recognition* (LNCS 2396, pp. 734-742). Berlin, Germany: Springer-Verlag.
- Pao, Y. H. (1989). *Adaptive pattern recognition and neural networks*. Reading MA: Addison-Wesley.
- Park, J., & Sandberg, I. W. (1993). Universal approximation using radial-basis-function networks. *Neural Computation*, 5, 305–316.
- Parveen, S. (2003). *Connectionist approaches to the deployment of prior knowledge for improving robustness in automatic speech recognition*. Unpublished doctoral dissertation, University of Sheffield.
- Parzen, E. (1962). On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, 33, 1065–1076. doi:10.1214/aoms/1177704472

- Pasula, H., Zettlemoyer, L., & Kaelbling, L. (2007). Learning symbolic models of stochastic domain. *Journal of Artificial Intelligence Research*, 29, 309–352.
- Pavesi, G., Mauri, G., & Pesole, G. (2001). An algorithm for finding signals of unknown length in DNA sequences. *Bioinformatics (Oxford, England)*, 17, S207–S214.
- Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 6(2), 559–572.
- Pelikan, E., De Groot, C., & Wurtz, D. (1992). Power consumption in West-Bohemia: Improved forecasts decorrelating connectionist networks. *Neural Network World*, 2(6), 701–712.
- Pelleg, D., & Moore, A. (1999). Accelerating exact k -means algorithms with geometric reasoning. In *Proceedings of the Fifth International Conference on Knowledge Discovery in Databases* (pp. 277-281). Menlo Park, CA: AAAI Press.
- Pennebaker, J., Francis, M., & Booth, R. (2001). *Linguistic inquiry and word count* (2nd ed.). New York: Psychology Press.
- Pernía-Espinoza, A. V., Castejón-Limas, M., González-Marcos, A., & Lobato-Rubio, V. (2005). Steel annealing furnace robust neural network model. *Ironmaking & Steelmaking*, 32(5), 418–426. doi:10.1179/174328105X28829
- Perrone, M. P., & Cooper, L. N. (1993). When networks disagree: Ensemble methods for hybrid neural networks. In R. J. Mammone (Ed.), *Neural networks for speech and image processing* (pp. 126-142). Boca Raton, FL: Chapman Hall.
- Petrovska-Delacrétaz, D., Chollet, G., & Dorizzi, B. (Eds.). (2008). *Guide to biometric reference systems and performance evaluation*. London: Springer-Verlag.
- Pevzner, P., & Sze, S. (2000). Combinatorial approaches to finding subtle signals in DNA sequences. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology* (pp. 269-278). Menlo Park, CA: AAAI Press.
- Phan, X.-H., Nguyen, L. M., & Horiguchi, S. (2008). *Learning to classify short and sparse text & Web with hidden topics from large-scale data collections*. Paper presented at the 17th International Conference on World Wide Web (WWW 2008).
- Phillips, P.J., Scruggs, W. T., O'Toole, A. J., Flynn, P. J., Bowyer, K. W., Schott, C. L., & Sharpe, M. (2007). *FRVT 2006 and ICE 2006 large-scale results*. NISTIR 7408.
- Piela, P. (2002). Introduction to self-organizing maps modelling for imputation - techniques and technology. *Research in Official Statistics*, 2, 5–19.
- Pike, J. (2003). *GPS overview from the NAVSTAR joint program office*. Retrieved from <http://www.fas.org/spp/military/program/nav/gps.htm>
- Piron, A., Leemans, V., Kleynen, O., Lebeau, F., & Destain, M. F. (2008). Selection of the most efficient wavelength bands for discriminating weeds from crop. *Computers and Electronics in Agriculture*, 62(2), 141–148. doi:10.1016/j.compag.2007.12.007
- Plamondon, L., Lapalme, G., & Pelletier, F. (2004). *Anonymisation de décisions de justice*. Paper presented at the XIe Conférence sur le Traitement Automatique des Langues Naturelles (TALN 2004).
- Platt, J. C. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in large margin classifiers* (pp. 61-74). Cambridge, MA: MIT Press.
- Plotkin, G. D. (1970). A note on inductive generalization. *Machine Intelligence*, 5, 153–163.
- Plumbley, M. (1997). Communications and neural networks: Theory and practice. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'97) - Volume 1* (p. 135).
- Poggio, T., & Girosi, F. (1990). Networks for approximation and learning, Networks for approximation and learning . *Proceedings of the IEEE*, 78, 1481–1497.
- Poh, N., & Kittler, J. (2007). On the use of log-likelihood ratio based model-specific score normalisation in biometric authentication. In S.-W. Lee & S. Z. Li (Eds.),

Compilation of References

- Advances in biometrics: Proc. of the ICB* (LNCS 4642, pp. 614–624). Berlin, Germany: Springer-Verlag.
- Poh, N., Bourlai, T., Kittler, J., Allano, L., Alonso, F., Ambekar, O., et al. (in press). Benchmarking quality-dependent and cost-sensitive multimodal biometric fusion algorithms. *IEEE Trans. Information Forensics and Security*.
- Polder, G., Van der Heijden, G. W. A. M., & Young, I. T. (2002). Spectral image analysis for measuring ripeness of tomatoes. *Transactions of the American Society of Agricultural Engineers*, 45(4), 1155–1161.
- Poli, I., & Jones, R. D. (1994). A neural net model for prediction. *Journal of the American Statistical Association*, 89(425), 117–121. doi:10.2307/2291206
- Pool, M. H., & Meuwissen, T. H. E. (1999). Prediction of daily milk yield from a limited number of test days using test day models. *Journal of Dairy Science*, 6, 1555–1564.
- Popescu, O., & Magnini, B. (2007). *Sense discriminative patterns for word sense disambiguation*. Paper presented at the Workshop on Semantic Content Acquisition and Representation (SCAR).
- Porter, M. (1980). An algorithm for suffix stripping. *Program*, 14(3), 130–137.
- Pouliquen, B., Kimler, M., Steinberger, R., Ignat, C., Oellinger, T., Blackler, K., et al. (2006). *Geocoding multilingual texts: Recognition, disambiguation and visualisation*. Paper presented at the 5th International Conference on Language Resources and Evaluation (LREC'2006).
- Powell, M. J. D. (1987). Radial basis functions for multivariable interpolation: a review, in Mason & Cox (Eds.), *Algorithms for Approximation*, Oxford: Clarendon Press.
- Pramodh, C., & Ravi, V. (2007). Modified great deluge algorithm based auto associative neural network for bankruptcy prediction in banks. *International Journal of Computational Intelligence Research*, 3(4), 363–370.
- Pratihar, D. K. (2003). Evolutionary robotics – a review. *Sadhan.*, 28(6), 999–1009. doi:10.1007/BF02703810
- Pressman, R. S. (1997). *Software engineering: A practitioner's approach* (4th ed.). New York: McGraw-Hill.
- Principe, J. C., de Vries, B., & de Oliveira, P. G. (1993). The gamma filter – a new class of adaptive IIR filters with restricted feedback. *IEEE Transactions on Signal Processing*, 41(2), 649–656. doi:10.1109/78.193206
- Proschan, M. A., McMahon, R. P., Shih, J. H., Hunsberger, S. A., Geller, N., Knatterud, G., & Wittes, J. (2001). Sensitivity analysis using an imputation method for missing binary data in clinical trials. *Journal of Statistical Planning and Inference*, 96(1), 155–165. doi:10.1016/S0378-3758(00)00332-3
- Provost, F., & Domingos, P. (2000). *Well-trained PETs: Improving probability estimation trees* (Tech. Rep. CDER #00-04-IS). Stern School of Business, New York University.
- Psarrou, A., Shaosheng, G., & Buxton, H. (1995, November). Modelling spatio-temporal trajectories and face signatures on partially recurrent neural networks. In *Proceedings of the IEEE International Conference on Neural Networks*, Perth, Australia (Vol. 5, pp. 2226–2231).
- Pydipati, R., Burks, T. F., & Lee, W. S. (2006). Identification of citrus disease using color texture features and discriminant analysis. *Computers and Electronics in Agriculture*, 52, 49–59. doi:10.1016/j.compag.2006.01.004
- Pyle, D. (1999). *Data preparation for data mining*. San Francisco: Morgan Kaufmann Publishers.
- Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, 1(1), 81–106.
- Quinlan, J. R. (1989). Unknown attribute values in induction. In Proceedings of the *International Workshop on Machine Learning* (pp. 164–168). San Francisco: Morgan Kaufmann Publishers Inc.
- Quinlan, J. R. (1993). *C4.5: Programs for machine learning*. San Francisco: Morgan Kaufmann.

- Quinlan, J. R., & Cameron-Jones, R. M. (1995). Introduction of logic programs: {FOIL} and related systems. *New Generation Computing*, 13, 287–312. doi:10.1007/BF03037228
- Rabiner, L. R. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2), 257–286. doi:10.1109/5.18626
- Rada, R. (2008). Expert systems and evolutionary computing for financial investing: A review. *Expert Systems with Applications*, 34(4), 2232–2240. doi:10.1016/j.eswa.2007.05.012
- Rada, R., & Ha, L. (2008). Intelligent technologies for investing: A review of engineering literature. *Intelligent Decision Technologies*, 2(3), 167–178.
- Ragg, T., Menzel, W., Baum, W., & Wigbers, M. (2002). Bayesian learning for sales rate prediction for thousands of retailers. *Neurocomputing*, 43, 127–144. doi:10.1016/S0925-2312(01)00624-5
- Rahimian, E., Singh, S., Thammachote, T., & Virmani, R. (1996). Bankruptcy prediction by neural network. In R. R. Trippi & E. Turban (Eds.), *Neural networks in finance and investing*. Burr Ridge, USA: Irwin Professional Publishing.
- Ramanathan, N., Chellappa, R., & Chowdhury, A. K. R. (2004). Facial similarity across age, disguise, illumination, and pose. In *Proc. of the Int. Conf. Image Processing* (pp. 1999-2002).
- Ramel, J. Y., Crucianu, M., Vincent, N., & Faure, C. (2003). Detection, extraction and representation of tables. In *Proceedings of the 7th International Conference on Document Analysis and Recognition* (pp. 374-378). Washington, DC: IEEE Computer Society.
- Ramon, J. (2002). *Clustering and instance based learning in first order logic*. Unpublished doctoral dissertation, K.U. Leuven, Belgium.
- Ramon, J., & Bruynooghe, M. (2001). A polynomial time computable metric between point sets. *Acta Informatica*, 37(10), 765–780. doi:10.1007/PL00013304
- Ramon, J., & Dehaspe, L. (1999). Upgrading Bayesian clustering to first order logic. In *Proceedings of the 9th Belgian-Dutch Conference on Machine Learning*, Department of Computer Science, K. U. Leuven.
- Rangarajan, S., Pboba, V., Balagani, K., Selmic, R., & Iyengar, S. (2004). Adaptive neural network clustering of Web users. *Computer*, 34–40. doi:10.1109/MC.2004.1297299
- Rangoni, Y., & Belaïd, A. (2006). Document logical structure analysis based on perceptive cycles. In H. Bunke, & A. L. Spitz (Eds.), *Proceedings of the 7th International Workshop on Document Analysis Systems* (pp. 117-128). Berlin, Germany: Springer.
- Rapid-I. (n.d.). *Rapid-I*. Retrieved from <http://yale.sf.net>
- Ravi Kumar, P., & Ravi, V. (2006). Bankruptcy prediction in banks by fuzzy rule based classifier. In *Proceedings of the 1st IEEE International Conference on Digital and Information Management*, Bangalore (pp. 222-227).
- Ravi Kumar, P., & Ravi, V. (2006). Bankruptcy prediction in banks by an ensemble classifier. In *Proceedings of the IEEE International Conference on Industrial Technology*, Mumbai (pp. 2032-2036).
- Ravi Kumar, P., & Ravi, V. (2007). Bankruptcy prediction in banks and firms via statistical and intelligent techniques - a review. *European Journal of Operational Research*, 180(1), 1–28. doi:10.1016/j.ejor.2006.08.043
- Ravi, V., & Pramodh, C. (2008). Threshold accepting trained principal component neural network and feature subset selection: Application to bankruptcy prediction in banks. *Applied Soft Computing*, 8(4), 1539–1548. doi:10.1016/j.asoc.2007.12.003
- Ravi, V., & Zimmermann, H. J. (2001). A neural network and fuzzy rule base hybrid for pattern classification. *Soft Computing*, 5(2), 152–159. doi:10.1007/s005000000071
- Ravi, V., & Zimmermann, H. J. (2003). Optimization of neural networks via threshold accepting: A comparison with back propagation algorithm. In *Proceedings of the Conference on Neuro-Computing and Evolving Intelligence*, Auckland, New Zealand.

Compilation of References

- Ravi, V., Kurniawan, H., Thai, P. N., & Ravi Kumar, P. (2008). Soft computing system for bank performance prediction. *Applied Soft Computing Journal*, 8(1), 305–315. doi:10.1016/j.asoc.2007.02.001
- Ravi, V., Murty, B. S. N., & Dutt, N. V. K. (2005). Forecasting the properties of carboxylic acids by a threshold accepting-trained neural network. *Indian Chemical Engineer*, 47(3), 147–151.
- Ravi, V., Ravi Kumar, P., Srinivas, E. R., & Kasabov, N. K. (2007). A semi-online training algorithm for the radial basis function neural networks: Applications to bankruptcy prediction in banks. In V. Ravi (Ed.), *Advances in banking technology and management: Impact of ICT and CRM*. Hershey, PA: IGI Global.
- Rawlings, J. O. (1988). *Applied regression analysis: A research tool*. CA: Wadsworth Inc.
- Reddy, R., & Ganguli, R. (2003). Structural damage detection in a helicopter rotor blade using radial basis function neural networks . *Smart Materials and Structures*, 12, 232–241.
- Reddy, R., & Tadepalli, P. (1997). Learning goal-decomposition rules using exercises. In *Proceedings of the International Conference on Machine Learning* (pp. 278–286).
- Redner, R. A., & Walker, H. F. (1984). Mixture densities, maximum likelihood, and the EM algorithm . *SIAM Review*, 26, 195–239.
- Reeve, R., Webb, B., Horchler, A., Indiveri, G., & Quinn, R. (2005). New technologies for testing a model of cricket phonotaxis on an outdoor robot platform. *Robotics and Autonomous Systems*, 51(1), 41–54. doi:10.1016/j.robot.2004.08.010
- Renders, J. M., & Flasse, S. P. (1998). Hybrid methods using genetic algorithms for global optimization. *IEEE Transactions on Systems . Man and Cybernetics Part-B*, 28(2), 73–91.
- Rendueles, J. L., González, J. A., Díaz, I., Diez, A., Seijo, F., & Cuadrado, A. (2006). Implementation of a virtual sensor on a hot dip galvanizing line for zinc coating thickness estimation. *Revue de Métallurgie-Cahiers D Informations Techniques*, 103(5), 226–232.
- Reyes-Sierra, M., & Coello, C. A. C. (2006). Multi-objective particle swarm optimizers: A survey of the state-of-the-art. *International Journal of Computational Intelligence Research*, 2(3), 287–308.
- Reynolds, D., Quatieri, T., & Dunn, R. (2000). Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10, 19–41. doi:10.1006/dspr.1999.0361
- Ribeiro, C. H. C. (1998). Embedding a priori knowledge in reinforcement learning. *Journal of Intelligent & Robotic Systems*, 21, 51–71. doi:10.1023/A:1007968115863
- Ridella, S., Rovetta, S., & Zunino, R. (1998). Plastic algorithm for adaptive vector quantization. *Neural Computing & Applications*, 7, 37–51. doi:10.1007/BF01413708
- Riedmiller, M. (1994). Supervised learning in multilayer perceptrons – from backpropagation to adaptive learning techniques. *International Journal on Computer Standards and Interfaces*, 16, 265–278. doi:10.1016/0920-5489(94)90017-5
- Rigamonti, M., Bloechle, J. L., Hadjar, K., Lalanne, D., & Ingold, R. (2005). Towards a canonical and structured representation of PDF documents through reverse engineering. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 1050-1055). Washington, DC: IEEE Computer Society.
- Rigoutsos, I., & Floratos, A. (1998). Combinatorial pattern discovery in biological sequences: The Teiresias algorithm. *Bioinformatics (Oxford, England)*, 14(1), 55–67. doi:10.1093/bioinformatics/14.1.55
- Rigoutsos, I., Floratos, A., Parida, L., Gao, Y., & Platt, D. (2000). The emergence of pattern discovery techniques in computational biology. *Metabolic Engineering*, 2(3), 159–177. doi:10.1006/mben.2000.0151
- Rijsbergen, K. van. (1979). *Information retrieval*. London: Butterworths.
- Riloff, E., Wiebe, J., & Wilson, T. (2003). *Learning subjective nouns using extraction pattern bootstrap*.

- ping. Paper presented at the 7th Conference on Natural Language Learning (CONLL-03).
- Ripley, B. D. (1995). *Pattern recognition and neural networks*. Cambridge, MA, USA: Cambridge University Press.
- Rissanen, J. (1986). Stochastic complexity and modeling. *Annals of Statistics*, 14(3), 1080–1100.
- Rissanen, J. (1989). Stochastic Complexity in Statistical Inquiry, World Scientific: Singapore.
- Ritter, H., Martinet, T., & Schulten, K. (1992). *Neural Computation and Self-Organizing maps: An introduction*. Reading, MA: Addison-Wesley.
- Rivest, R. L. (1987). Learning Decision List. *Machine Learning*, 2, 229–246.
- Rockafellar, R. (1972), Convex Analysis, Princeton University Press.
- Rosenblatt, F. (1962). *Principles of neurodynamics*. New York: Spartan.
- Rosenbloom, P. S., Newell, A., & Lairdmon, J. E. (1993). Towards the knowledge level in Soar: The role of the architecture in the use of knowledge. *The Soar papers: Research on integrated intelligence*, 2, 897-933.
- Ross, A., & Jain, A. (2003). Information fusion in biometrics. *Pattern Recognition Letters*, 24, 2115–2125. doi:10.1016/S0167-8655(03)00079-5
- Rouveiro, C. (1992). Extensions of inversion of resolution applied to theory completion. In *Inductive logic programming* (pp. 64-90). Amsterdam: Academic Press.
- Roweis, S. (1998). EM algorithms for PCA and SPCA. In *Advances in neural information processing systems* (pp. 626-632). Cambridge, MA: MIT Press.
- Roweis, S., & Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290, 2323–2326. doi:10.1126/science.290.5500.2323
- Rubin, D. B. (1987). *Multiple imputation for nonresponse in surveys*. New York: Wiley.
- Ruiz, F. A., Castel, J. M., Mena, Y., Camuñez, J., & González-Redondo, P. (2008). Application of the technico-economic analysis for characterizing,making diagnoses and improving pastoral dairy goatsystems in Andalusia (Spain). *Small Ruminant Research*, 77, 208–220. doi:10.1016/j.smallrumres.2008.03.007
- Rumelhart, D. E., Hinton, G. E., & Williams, R. S. (1986). Learning representations by back-propagating errors. *Nature*, 323, 533–536. doi:10.1038/323533a0
- Rumelhart, D., & Zipser, D. (1985). Feature discovery by competitive learning. *Cognitive Science*, 9, 75–112.
- Runeson, P., & Wohlin, C. (1998). Defect content estimations from review data. In *Proceedings of the 20th International Conference on Software Engineering* (pp. 400-409).
- Rupérez, M. J., Monserrat, C., & Alcañiz, M. (2008). Simulation of the deformation of materials in shoe uppers in gait. Force distribution using finite elements. *Int J Interact Des Manuf Biomech*, 2, 59–68. doi:10.1007/s12008-008-0036-6
- Sadlier, D., & Connor, N. (2005). Event detection in field-sports video using audio-visual features and a support vector machine. *IEEE Trans. Circuits Syst. Video Technology*, 15(10), 1225–1233. doi:10.1109/TCSVT.2005.854237
- Sagot, M.-F. (1998). Spelling approximate repeated or common motifs using a suffix tree. In *Theoretical informatics* (LNCS 1380, pp. 111-127). Berlin, Germany: Springer.
- Sahai, S., & Pahwa, A. (2004). Failures of overhead distribution system lines due to animals. In *Proceedings of North American Power Symposium*, Moscow, Idaho.
- Salah, A. A., & Alpaydin, E. (2004). Incremental mixtures of factor analysers. In *Proc. of the Int. Conf. on Pattern Recognition* (pp. 276-279).
- Salah, A. A., Alpaydin, E., & Akarun, L. (2002). A selective attention-based method for visual pattern recognition with application to handwritten digit recognition and face recognition. *IEEE Transactions on Pattern*

Compilation of References

- Analysis and Machine Intelligence*, 24(3), 420–425. doi:10.1109/34.990146
- Salah, A. A., Çınar, H., Akarun, L., & Sankur, B. (2007). Robust facial landmarking for registration. *Annales des Télécommunications*, 62(1-2), 1608–1633.
- Salchenberger, L. C., & Lash, N. (1992). Neural networks, mine: A tool for predicting thrift failures. *Decision Sciences*, 23, 899–916. doi:10.1111/j.1540-5915.1992.tb00425.x
- Samad, T., & Harp, S. A. (1992). Self-organization with partial data. *Network*, 3, 205–212.
- Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Transactions on Computers*, 18(5), 401–409. doi:10.1109/T-C.1969.222678
- Samson, B., Ellison, D., & Dugard, P. (1993). Software costestimation using an Albus perceptron. In *Proceedings of the 8th International COCOMO Estimation meeting*, Pittsburgh, USA.
- Samuelsson, C. (2004). Statistical methods. In R. Mitkov (Ed.), *The Oxford handbook of computational linguistics* (pp. 358-375). Oxford, UK: Oxford University Press.
- Sancho-Gómez, J. L., García-Laencina, P. J., & Figueiras-Vidal, A. R. (2008). Combining missing data imputation and classification in a multi-layer perceptron. *Intelligent Automation and Soft Computing*.
- Sanders, F. (1963). On subjective probability forecasting. *Journal of Applied Meteorology*, 2, 191–201. doi:10.1175/1520-0450(1963)002<0191:OSPF>2.0.CO;2
- Sandve, G., & Drabløs, F. (2006). A survey of motif discovery methods in an integrated framework. *Biology Direct*, 1(11).
- Sankar, K. P., Ambati, V., Pratha, L., & Jawahar, C. V. (2006). Digitizing a million books: Challenges for document analysis. In H. Bunke & A. L. Spitz (Eds.), *Proceedings of the 7th International Workshop on Document Analysis Systems* (pp. 425-436). Berlin, Germany: Springer.
- Sanzogni, L., & Kerr, D. (2001). Milk production estimates using feed forward artificial neural networks. *Computers and Electronics in Agriculture*, 32, 21–30. doi:10.1016/S0168-1699(01)00151-X
- Sarafijanovic, S., & Le Boudec, J.-Y. (2005). An artificial immune system approach with secondary response for misbehavior detection in mobile ad hoc networks. *IEEE Transactions on Neural Networks*, 16(5), 1076–1087. doi:10.1109/TNN.2005.853419
- Sardari, S., & Sardari, D. (2002). Applications of artificial neural network in AIDS research and therapy. *Current Pharmaceutical Design*, 8(8), 659–670. doi:10.2174/1381612024607199
- Sarimveis, H., Doganis, P., & Alexandridis, A. (2006). classification technique based on radial basis function neural networks . *Advances in Engineering Software*, 37(4), 218–221.
- Schaffer, J. L. (1997). *Analysis of incomplete multivariate data*. Boca Raton, FL: Chapman & Hall.
- Schiefer, C., Jörgl, H. P., Rubenzucker, F. X., & Aberl, H. (1999). Adaptive control of a galvannealing process with a radial basis function network. In *Proc. of the 14th IFAC World Congress, Vol. 0*, Beijing, PR China (pp. 61-66).
- Schiffmann, W., Joost, M., & Werner, R. (1993). Application of genetic algorithms to the construction of topologies for multi layer perceptron. In *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms* (pp. 675-682).
- Schikora, P. F., & Godfrey, M. R. (2003). Efficacy of end-user neural network and data mining software for predicting complex system performance. *International Journal of Production Economics*, 84(3), 231–253.
- Schofield, C. (1998). *Non-algorithmic effort estimation techniques* (Tech. Rep. TR98-01).
- Schölkopf, B., Herbrich, R., & Smola, A. J. (2001). A generalized representer theorem. In *Proceedings of the 14th Annual Conference on Computational Learning Theory and 5th European Conference on Computational*

- Learning Theory (COLT '01/EuroCOLT '01)* (pp. 416-426). London, UK: Springer-Verlag.
- Schölkopf, B., Mika, S., Burges, C., Knirsch, P., Müller, K., Rätsch, G., & Smola, A. (1999). Input space versus feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5), 1000–1017. doi:10.1109/72.788641
- Schölkopf, B., Smola, A. J., & Müller, K.-R. (1997). Kernel principal component analysis. In *Proceedings of the ICANN* (pp. 583-588).
- Schölkopf, B., Smola, A., & Müller, K. R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5), 1299–1319. doi:10.1162/089976698300017467
- Schöllhorn, W. I. (2004). Applications of artificial neural nets in clinical biomechanics. *Clinical Biomechanics (Bristol, Avon)*, 19, 876–898. doi:10.1016/j.clinbiomech.2004.04.005
- Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics*, 6, 461–464.
- Sebag, M. (1997). Distance induction in first order logic. In N. Lavrač & S. Džeroski (Eds.), In *Proceedings of the ILP-97*. Berlin, Germany: Springer.
- Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1), 1–47. doi:10.1145/505282.505283
- Seki, M., Fujio, M., Nagasaki, T., Shinjo, H., & Marukawa, K. (2007). Information management system using structure analysis of paper/electronic documents and its applications. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 689-693). Washington, DC: IEEE Computer Society.
- Seluca, C. (1995). *An investigation into software effort estimation using a back-propagation neural network*. Unpublished master's thesis, Bournemouth University, UK.
- Semeraro, G., Esposito, F., Malerba, D., Fanizzi, N., & Ferilli, S. (1998). A logic framework for the incremental inductive synthesis of datalog theories. In N. E. Fuchs (Ed.), *Logic program synthesis and transformation* (LNCS 1463, pp. 300-321). Berlin, Germany: Springer.
- Sevag, M. (1997). Distance induction in first order logic. In *Proceedings of the 7th International Workshop on Inductive Logic Programming* (pp. 264-272).
- Shahin, M. A., Tollner, E. W., Gitaitis, R. D., Sumner, D. R., & Maw, B. W. (2002). Classification of sweet onions based on internal defects using image processing and neural network techniques. *Transactions of the American Society of Agricultural Engineers*, 45(5), 1613–1618.
- Shaosheng, F., & Hui, X. (2004). Diagonal recurrent neural network based predictive control for active power filter. In *Proceedings of the International Conference on Power Systems Technology, POWERCON 2004*, Singapore (pp. 759-762).
- Sharpe, I. G., & Kofman, P. (2003). Using multiple imputation in the analysis of incomplete observations in finance. *Journal of Financial Econometrics*, 1(2), 216–249. doi:10.1093/jjfinec/nbg013
- Sharpe, P. K., & Solly, R. J. (1995). Dealing with missing values in neural network-based diagnostic systems. *Neural Computing & Applications*, 3(2), 73–77. doi:10.1007/BF01421959
- Shawe-Taylor, J., & Christianini, N. (2004). *Kernel methods for pattern analysis*. Cambridge, UK: Cambridge University Press.
- Shearer, S. A., & Payne, F. A. (1990). Color and defect sorting of bell peppers using machine vision. *Transactions of the American Society of Agricultural Engineers*, 33(6), 2045–2050.
- Shen, W. M., & Simon, H. A. (1989). Rule creation and rule learning through environmental exploration. In, *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 675-680).
- Sherriff, M., Nagappan, N., Williams, L., & Vouk, M. (2005). Early estimation of defect density using an in-process Haskell metrics model. *Software Engineering Notes*, 30(4), 1–6. doi:10.1145/1082983.1083285

Compilation of References

- Shi, L. (2008). Bayesian Ying-Yang harmony learning for local factor analysis: a comparative investigation. In Tizhoosh & Ventresca (eds), *Oppositional Concepts in Computational Intelligence*, Springer-Verlag, 209-232.
- Shi, Z., & Govindaraju, V. (2005). Multi-scale techniques for document page segmentation. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 1020-1024). Washington, DC: IEEE Computer Society.
- Shin, K. S., Lee, T. S., & Kim, H. J. (2005). An application of support vector machines in bankruptcy prediction model. *Expert Systems with Applications*, 28, 127–135. doi:10.1016/j.eswa.2004.08.009
- Shtub, A., & Versano, R. (1999). Estimating the cost of steel pipe bending, a comparison between neural networks and regression analysis. *International Journal of Production Economics*, 62(3), 201–207. doi:10.1016/S0925-5273(98)00212-6
- Siddiqi, A., & Lucas, S. (1998). A comparison of matrix rewriting versus direct encoding for evolving neural networks. In *Proceedings of the IEEE Int. Conf. on Evolutionary Computation* (pp. 392-397).
- Silver, D. L. (2000). *Selective transfer of neural network task knowledge*. Unpublished doctoral dissertation, University of Western Ontario.
- Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. Boca Raton, FL: Chapman and Hall.
- Simon, G., Lee, J., & Verleysen, M. (2006). Unfolding preprocessing for meaningful time series clustering. *Neural Networks*, 19(6-7), 877–888. doi:10.1016/j.neunet.2006.05.020
- Sinzinger, E. D., & Moore, B. (2005). Sedation of simulated ICU patients using reinforcement learning based control. *International Journal of Artificial Intelligence Tools*, 14(1-2), 137–156. doi:10.1142/S021821300500203X
- Sipser, M. (1997). *Introduction to the theory of computation*. Boston, MA: PWS Publishing.
- Sleator, D., & Temperley, D. (1993). *Parsing English with a link grammar*. Paper presented at the Third International Workshop on Parsing Technologies.
- Smola, A. J., Mika, S., Schölkopf, B., & Williamson, R. C. (2001). Regularized principal manifolds. *Journal of Machine Learning Research*, 1, 179–209. doi:10.1162/15324430152748227
- Sokolova, M., & Lapalme, G. (2007). *Performance measures in classification of human communication*. Paper presented at the 20th Canadian Conference on Artificial Intelligence (AI'2007).
- Sokolova, M., Nastase, V., & Szpakowicz, S. (2008). *The telling tail: Signals of success in electronic negotiation texts*. Paper presented at the Third International Joint Conference on Natural Language Processing (IJCNLP 2008).
- Song, Q., Shepperd, M. J., Cartwright, M., & Mair, C. (2006). Software defect association mining and defect correction effort prediction. *IEEE Transactions on Software Engineering*, 32(2), 69–82. doi:10.1109/TSE.2006.1599417
- Specht, D. F. (1988). Probabilistic neural networks for classification, mapping, or associative memory. In *Proceedings of the IEEE Conference on Neural Networks*, 1, 525–532. doi:10.1109/ICNN.1988.23887
- Specht, D. F. (1990). Probabilistic neural networks. *Neural Networks*, 3(1), 109–118. doi:10.1016/0893-6080(90)90049-Q
- Specht, D. F. (1991). A general regression neural network. *IEEE Transactions on Neural Networks*, 2(6), 568–576. doi:10.1109/72.97934
- Srinivasan, K., & Fisher, D. (1995). Machine learning approaches to estimating software development effort. *IEEE Transactions on Software Engineering*, 21(2), 126–136. doi:10.1109/32.345828
- Srinivasan, M., Venkatesh, S., & Hosie, R. (1997). Qualitative estimation of camera motion parameters from video sequences. *Pattern Recognition*, 30, 593–606. doi:10.1016/S0031-3203(96)00106-9

- Stauffer, C., & Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 246-252).
- Steinbach, M., Karypis, G., & Kumar, V. (2000). A comparison of document clustering techniques. *Proceedings of KDD Workshop on Text Mining*.
- Steinberger, R., & Pouliquen, B. (2007). Cross-lingual named entity recognition. *Linguisticae Investigationes*, 30(1), 135–162.
- Steinberger, R., Fuart, F., van der Goot, E., Best, C., von Etter, P., & Yangarber, R. (2008). Text mining from the Web for medical intelligence. In D. Perrotta, J. Piskorski, F. Soulié-Fogelman, & R. Steinberger (Eds.), *Mining massive data sets for security*. Amsterdam: OIS Press.
- Steinberger, R., Pouliquen, B., Widiger, A., Ignat, C., Ejavec, T., Tufis, D., & Varga, D. (2006). *The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages*. Paper presented at the 5th International Conference on Language Resources and Evaluation (LREC 2006).
- Steinhaus, H. (1956). Sur la division des corps matériels en parties. In *Bull. Acad. Polon. Sci., Cl. III vol IV* (pp. 801-804).
- Stenning, K., Lascarides, A., & Calder, J. (2006). *Introduction to cognition and communication*. Cambridge, MA: The MIT Press.
- Stevenson, M., & Wilks, Y. (2001). The interaction of knowledge sources in word sense disambiguation. *Computational Linguistics*, 27(3), 321–349. doi:10.1162/089120101317066104
- Stockwell, R. G., Mansinha, L., & Lowe, R. P. (1996). Localization of the complex spectrum: The S-transform. *IEEE Transactions on Signal Processing*, 44(4), 998–1001. doi:10.1109/78.492555
- Stokes, N., Li, Y., Moffat, A., & Rong, J. (2008). An empirical study of the effects of NLP components on geographic IR performance. *International Journal of Geographical Information Science*, 22(3), 247–264. doi:10.1080/13658810701626210
- Stolcke, A., & Omohundro, S. (1993). Hidden Markov model induction by Bayesian model merging. In S. J. Hanson, J. D. Cowan, & C. L. Giles (Eds.), *Advances in neural information processing systems, vol. 5* (pp. 11-18). San Francisco, CA: Morgan Kauffmann.
- Stone, M. (1978). Cross-validation: A review . *Math. Operat. Statist.*, 9, 127–140.
- Storn, R., & Price, K. (1995). *Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces* (Tech. Rep. No. TR-95-012). International Computer Science Institute, Berkeley.
- Strackeljan, J., & Leiviskä, K. (2008). Artificial immune system approach for the fault detection in rotating machinery. In Proceedings of the International Conference on Condition Monitoring & Machinery Failure Prevention Technologies, Edinburgh, UK.
- Strang, G. (1993). The fundamental theorem of linear algebra. *The American Mathematical Monthly*, 100(9), 848–855. doi:10.2307/2324660
- Strapparava, C., Valitutti, A., & Stock, O. (2006). *The affective weight of the lexicon*. Paper presented at the 5th International Conference on Language Resources and Evaluation (LREC 2006).
- Strickert, M., & Hammer, B. (2005). Merge SOM for temporal data. *Neurocomputing*, 64, 39–71. doi:10.1016/j.neucom.2004.11.014
- Strukov, D. B., Snider, G. S., Stewart, D. R., & Williams, R. S. (2008, May). The missing memristor found. *Nature*, 453, 80–83. doi:10.1038/nature06932
- Sugeno, M. (1985). An introductory survey of fuzzy control. *Information Sciences*, 36(1-2), 59–83. doi:10.1016/0020-0255(85)90026-X
- Sun, K., Tu, S. K., Gao, D. Y., & Xu, L. (2009), Canonical Dual Approach to Binary Factor Analysis, To appear in Proc. 8th International Conf on Independent Component Analysis and Signal Separation, ICA 2009, Paraty, Brazil, March 15-18, 2009.
- Sun, Z. L., Au, K. F., & Choi, T. M. (2007). A hybrid neuron-fuzzy inference system through integration of fuzzy

Compilation of References

- logic and extreme learning machines. *IEEE Transactions on Systems, Man, and Cybernetics. Part B, Cybernetics*, 37(5), 1321–1331. doi:10.1109/TSMCB.2007.901375
- Sun, Z. L., Choi, T. M., Au, K. F., & Yu, Y. (2008). Sales forecasting using extreme learning machine with applications in fashion retailing. *Decision Support Systems*, 46, 411–419. doi:10.1016/j.dss.2008.07.009
- Sun, Z. L., Huang, D. S., Zheng, C. H., & Shang, L. (2006). Optimal selection of time lags for TDSEP based on genetic algorithm. *Neurocomputing*, 69(7-9), 884–887. doi:10.1016/j.neucom.2005.06.010
- Sun, Z., Wang, Y., Tan, T., & Cui, J. (2005). Improving iris recognition accuracy via cascaded classifiers. *IEEE Trans. Systems, Man, and Cybernetics. Part C: Applications and Reviews*, 35(3), 435–441. doi:10.1109/TSMCC.2005.848169
- Surdeanu, M., Turmo, J., & Comelles, E. (2005). *Named entity recognition from spontaneous open-domain speech*. Paper presented at the Interspeech 2005–Europespeech.
- Sutton, R. S., & Barto, A. (1998). *Reinforcement learning. An introduction*. Cambridge, MA: The MIT Press.
- Sweeney, L. (1996). Replacing personally-identifying information in medical records, the scrub system. *Journal of the American Medical Informatics Association*, 333–337.
- Sylvester, J. J. (1889). On the reduction of a bilinear quantic of the n th order to the form of a sum of n products by a double orthogonal substitution. *Messenger of Mathematics*, 19, 42–46.
- Szarvas, G., & Busa-Fekete, R. (2006). *State-of-the-art anonymization of medical records using an iterative machine learning framework*. Paper presented at the i2b2 Workshop on Challenges in Natural Language Processing for Clinical Data.
- Sztandera, L. M., Frank, C., & Vemulapali, B. (2004). Prediction of women's apparel sales using soft computing methods. In *Knowledge-based intelligent information and engineering systems* (LNCS 3214, pp. 506–512). Berlin, Germany: Springer.
- Takagi, T., & Sugeno, M. (1985). Fuzzy identification of systems and its application to modeling and control. *IEEE Transactions on Systems, Man, and Cybernetics*, 15, 116–132.
- Tam, K. Y. (1991). Neural network models and the prediction of bank bankruptcy. *Omega*, 19, 429–445. doi:10.1016/0305-0483(91)90060-7
- Tan, Z., & Zhang, J. (2008). An empirical study of sentiment analysis for Chinese documents. *Expert Systems with Applications*, 34(4), 2622–2629. doi:10.1016/j.eswa.2007.05.028
- Tang, Z. Y., Almeida, C., & Fishwick, P. A. (1991). Time series forecasting using neural networks vs. Box- Jenkins methodology. *Simulation*, 57, 303–310. doi:10.1177/003754979105700508
- Tao, Y., Heinemann, P. H., Varghese, Z., Morrow, C. T., & Sommer, H. J. III. (1995). Machine vision for color inspection of potatoes and apples. *Transactions of the American Society of Agricultural Engineers*, 38(5), 1555–1561.
- Tarpey, T., & Flury, B. (1996). Self-consistency: A fundamental concept in statistics. *Statistical Science*, 11(3), 229–243. doi:10.1214/ss/1032280215
- Taylor, D. W., & Corne, D. W. (2003). An investigation of the negative selection algorithm for fault detection in refrigeration systems. In *Artificial immune systems* (LNCS 2787, pp. 34–45). Berlin, Germany: Springer.
- Teh, H.-H. (1995). *Neural logic networks*. Singapore: World Scientific.
- Tenenbaum, J. B., de Silva, V., & Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *Science*, 290, 2319–2323. doi:10.1126/science.290.5500.2319
- Tenner, J., Linkens, D. A., Morris, P. F., & Bailey, T. J. (2001). Prediction of mechanical properties in steel heat treatment process using neural networks. *Ironmaking & Steelmaking*, 28(1), 15–22. doi:10.1179/030192301677803

- Teo, J. (2005). Evolutionary multi-objective optimization for automatic synthesis of artificial neural network robot controllers. *Malaysian Journal of Computer Science*, 18(2), 54–62.
- Teo, J., & Abbass, H. A. (2004). Embodied legged organisms: A Pareto evolutionary multi-objective approach. *Journal of Evolutionary Computation*, 12(3), 355–394. doi:10.1162/1063656041774974
- Teo, J., & Abbass, H. A. (2004). Multi-objectivity and complexity in embodied cognition. *IEEE Transactions on Evolutionary Computation*, 9(4), 337–360. doi:10.1109/TEVC.2005.846902
- Thimm, G., & Fiesler, E. (1995). Evaluating pruning methods. In *Proc. of the International Symposium on Artificial Neural Networks* (pp. 20-25).
- Thomas, M., Pang, B., & Lee, L. (2006). *Get out the vote: Determining support or opposition from congressional floor-debate transcripts*. Paper presented at the Empirical Methods in Natural Language Processing (EMNLP 2006).
- Thomas, T. J. (2000). *Locally-connected neural network architecture for invariant pattern recognition* (NPO-20633). USA: NASA Technical Briefings.
- Thomassey, S., Happiette, M., & Castelain, J. M. (2005). A global forecasting support system adapted to textile distribution. *International Journal of Production Economics*, 96(1), 81–95. doi:10.1016/j.ijpe.2004.03.001
- Thompson, K., & Langley, P. (1989). Incremental concept formation with composite objects. In *Proceedings of the 6th International Workshop on Machine Learning*. San Francisco: Morgan Kaufmann.
- Thrun, S. (1992). The role of exploration in learning control. In D. A. White & D. Sofge (Eds.), *Handbook of intelligent control: Neural, fuzzy, and adaptive approaches*. Florence, KY: Van Nostrand Reinhold.
- Tian, Y. C., Hou, C. H., & Gao, F. (2000). Mathematical modelling of a continuous galvanizing annealing furnace. *Developments in Chemical Engineering & Mineral Processing*, 8(3/4), 359–374.
- Tibshirani, R. (1992). Principal curves revisited. *Statistics and Computing*, 2, 183–190. doi:10.1007/BF01889678
- Tickle, A. B., Andrews, R., Golea, M., & Diederich, J. (1998). The truth will come to light: Directions and challenges in extracting the knowledge embedded within trained artificial neural network. *IEEE Transactions on Neural Networks*, 9(6), 1057–1068. doi:10.1109/72.728352
- Till, M., & Rudolph, S. (2000). Optimized time-frequency distributions for signal classification with feed-forward neural networks. In [Orlando: SPIE Proceedings Series.]. *Proceedings of the Applications and Science of Computational Intelligence III*, 4055, 299–310.
- Tiňo, P., & Nabney, I. (2002). Hierarchical GTM: Constructing localized nonlinear projection manifolds in a principled way. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5), 639–656. doi:10.1109/34.1000238
- Tiňo, P., Farkaš, I., & van Mourik, J. (2006). Dynamics and topographic organization of recursive self-organizing maps. *Neural Computation*, 18, 2529–2567. doi:10.1162/neco.2006.18.10.2529
- Tistarelli, M., Bicego, M., Alba-Castro, J. L., González-Jiménez, D., Mellakh, A., Salah, A. A., et al. (2008). 2D face recognition. In D. Petrovska-Delacrétaz, G. Chollet, & B. Dorizzi (Eds.), *Guide to biometric reference systems and performance evaluation* (pp. 217-266). London: Springer-Verlag.
- Tompa, M. (2005). Assessing computational tools for the discovery of transcription factor binding sites. *Nature Biotechnology*, 23(1), 137–144. doi:10.1038/nbt1053
- Towell, G., & Shavlik, J. (1993). The extraction of refined rules from knowledge based neural network. *Machine Learning*, 131, 71–101.
- Trajanoski, Z., Regitnig, W., & Wach, P. (1998). Simulation studies on neural predictive control of glucose using the subcutaneous route. *Computer Methods and Programs in Biomedicine*, 56(2), 133–139. doi:10.1016/S0169-2607(98)00020-0

Compilation of References

- Treier, S., & Jackman, S. (2002). Beyond factor analysis: modern tools for social measurement. Presented at the 2002 Annual Meetings of the Western Political Science Association and the Midwest Political Science Association.
- Tresp, V., Ahmad, S., & Neuneier, R. (1993). Training neural networks with deficient data. In J. D. Cowan, et al. (Eds.), *Advances on neural information processing systems 6* (pp. 128-135). San Francisco: Morgan Kaufmann Publishers Inc.
- Tresp, V., Neuneier, R., & Ahmad, S. (1994). Efficient methods for dealing with missing data in supervised learning. In G. Tesauro, et al. (Eds.), *Advances on neural information processing systems 7* (pp. 689-696). Cambridge, MA: The MIT Press.
- Troyanskaya, O., Cantor, M., Alter, O., Sherlock, G., Brown, P., & Botstein, D. (2001). Missing value estimation methods for DNA microarrays. *Bioinformatics (Oxford, England)*, 17(6), 520–525. doi:10.1093/bioinformatics/17.6.520
- Tsakonas, A., Dounias, G., Doumpas, M., & Zopounidis, C. (2006). Bankruptcy prediction with neural logic networks by means of grammar-guided genetic programming. *Expert Systems with Applications*, 30(3), 449–461. doi:10.1016/j.eswa.2005.10.009
- Tsoi, A. C., & Back, A. D. (1994). Locally recurrent globally feedforward networks: A critical review of architectures. *IEEE Transactions on Neural Networks*, 5(2), 229–239. doi:10.1109/72.279187
- Tsoi, A. C., & Back, A. D. (1997). Discrete time recurrent neural network architectures: A unifying review. *Neurocomputing*, 15(3-4), 183–223. doi:10.1016/S0925-2312(97)00161-6
- Tuganbaev, D., Pakhchanian, A., & Deryagin, D. (2005). Universal data capture technology from semi-structured forms. In *Proceedings of the 8th International Conference on Document Analysis and Recognition* (pp. 458-462). Washington, DC: IEEE Computer Society.
- Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1), 71–86. doi:10.1162/jocn.1991.3.1.71
- Turney, P., & Littman, M. (2003). Measuring praise and criticism: Inference of semantic orientation from association. *ACM Transactions on Information Systems*, 21(4), 315–346. doi:10.1145/944012.944013
- Tversky, A. (1977). Features of similarity. *Psychological Review*, 84(4), 327–352. doi:10.1037/0033-295X.84.4.327
- Uchiyama, T., Shimohara, K., & Tokunaga, Y. (1989). A modified leaky integrator network for temporal pattern recognition. In *Proceedings of the International Joint Conference on Neural Networks*, 1, 469–475. doi:10.1109/IJCNN.1989.118621
- Ukkonen, E., Vilo, J., Brazma, A., & Jonassen, I. (1996). Discovering patterns and subfamilies in biosequences. In *Proceedings of the 4th International Conference on Intelligent Systems for Molecular Biology* (pp. 34-43). Menlo Park, CA: AAAI Press.
- Uncini, A. (2002). Sound synthesis by flexible activation function recurrent neural networks. In M. Marinaro & R. Tagliaferri (Eds.), *Proceedings of the WIRN VIETRI 2002* (LNCS 2486, pp. 168-177).
- Urzelai, J., & Floreano, D. (2000). Evolutionary robots with fast adaptive behavior in new environments. In *Proceedings of the Third International Conference on Evolvable Systems: From Biology to Hardware (ICES2000)*, Edinburgh, Scotland.
- Uzuner, O., Luo, Y., & Szolovits, P. (2007). Evaluating the state-of-the-art in automatic de-indentification. *Journal of the American Medical Informatics Association*, 14, 550–563. doi:10.1197/jamia.M2444
- Uzuner, O., Sibanda, T., Luo, Y., & Szolovits, P. (2008). A de-identifier for medical discharge summaries. *Journal of Artificial Intelligence in Medicine*, 42, 13–35. doi:10.1016/j.artmed.2007.10.001
- van Beusekom, J., Keysers, D., Shafait, F., & Breuel, T. M. (2006). Distance measures for layout-based document image retrieval. In *Proceedings of the 2nd International Workshop on Document Image Analysis for Libraries* (pp. 232-242). Washington, DC: IEEE Computer Society.

- van Beusekom, J., Keysers, D., Shafait, F., & Breuel, T. M. (2007). Example-based logical labeling of document title page images. In *Proceedings of the 9th International Conference on Document Analysis and Recognition* (pp. 919-923). Washington, DC: IEEE Computer Society.
- van Lent, M., & Laird, J. (2001). Learning procedural knowledge through observation. In *Proceedings of the International conference on Knowledge Capture*.
- Vander Wiel, S. A., & Votta, L. G. (1993). Assessing software designs using capture-recapture methods. *IEEE Transactions on Software Engineering*, 19(11), 1045–1054. doi:10.1109/32.256852
- Vapnik, V. (1995). *The nature of statistical learning theory*. New York: Springer-Verlag.
- Vapnik, V. (2000). *The nature of statistical learning theory* (2nd ed.). New York: Springer-Verlag.
- Vapnik, V. N. (1998). *Statistical learning theory*. New York: John Wiley & Sons.
- Varasai, P., Pechsiri, C., Sukvari, T., Satayamas, V., & Kawtrakul, A. (2008). *Building an annotated corpus for text summarization and question answering*. Paper presented at the Sixth International Language Resources and Evaluation (LREC'08).
- Vellido, A. (2006). Missing data imputation through GTM as a mixture of *t*-distributions. *Neural Networks*, 19(10), 1624–1635. doi:10.1016/j.neunet.2005.11.003
- Vellido, A., & Lisboa, P. J. G. (2006). Handling outliers in brain tumour MRS data analysis through robust topographic mapping. *Computers in Biology and Medicine*, 36(10), 1049–1063. doi:10.1016/j.compbiomed.2005.09.004
- Vellido, A., El-Deredy, W., & Lisboa, P. J. G. (2003). Selective smoothing of the generative topographic mapping. *IEEE Transactions on Neural Networks*, 14(4), 847–852. doi:10.1109/TNN.2003.813834
- Veloso, M., & Carbonell, J. (1993). Derivational analogy in PRODIGY: Automating case acquisition, storage, and utilization. *Machine Learning*, 10, 249–278. doi:10.1023/A:1022686910523
- Veloso, M., Carbonell, J., Pérez, A., Borrajo, D., Fink, E., & Blythe, J. (1995). Integrating planning and learning: The PRODIGY architecture. *JETAI*, 7(1), 81–120. doi:10.1080/09528139508953801
- Venkatachalam, A. R. (1993). Software cost estimation using artificial neural networks. In *Proceedings of the International Joint Conference on Neural networks*, Nagoya (pp. 987-990). Washington DC: IEEE.
- Verbeek, J. J., Vlassis, N., & Kröse, B. (2002). A *k*-segments algorithm for finding principal curves. *Pattern Recognition Letters*, 23(8), 1009–1017. doi:10.1016/S0167-8655(02)00032-6
- Verdejo, R., & Mills, N. J. (2004). Heel-Shoe interactions and the durability of EVA foam running-shoe midsoles. *Journal of Biomechanics*, 37, 1379–1386. doi:10.1016/j.jbiomech.2003.12.022
- Vinaykumar, K., Ravi, V., Carr, M., & Raj Kiran, N. (2008). Software development cost estimation using wavelet neural networks. *Journal of Systems and Software*, 81(11), 1853–1867. doi:10.1016/j.jss.2007.12.793
- Vinokourov, A., Shawe-Taylor, J., & Cristianini, N. (2002). *Inferring a semantic representation of text via cross-language correlation analysis*. Paper presented at the Neural Information Processing Systems (NIPS 2002).
- Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.
- Voegtlin, T. (2002). Recursive self-organizing maps. *Neural Networks*, 15(8-9), 979–991. doi:10.1016/S0893-6080(02)00072-2
- von der Malsburg, C. (1973). Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14, 85–100. doi:10.1007/BF00288907
- Wahba, G., Lin, Y., & Zhang, H. (2000). Generalized approximate cross validation for support vector machines. In A. Smola, P. Bartlett, B. Schölkopf, & C. Schuurmans (Eds.), *Advances in large margin classifiers* (pp. 297-309). Cambridge, MA: MIT Press.

Compilation of References

- Wallace, C. S., & Boulton, D. M. (1968). An information measure for classification . *The Computer Journal*, 11, 185–194.
- Wallace, C., & Freeman, P. (1987). Estimation and inference by compact coding . *Journal of the Royal Statistical Society. Series A (General)*, 49(3), 240–265.
- Wan, E. (1993). Time series prediction using a neural network with embedded tapped delay-lines. In A. Weigend & N. Gershenfeld (Eds.), *Predicting the future and understanding the past, SFI studies in the science of complexity*. Reading, MA: Addison-Wesley.
- Wang, J.-L., & Chan, S.-H. (2006). Stock market trading rule discovery using two-layer bias decision tree. *Expert Systems with Applications*, 30(4), 605–611. doi:10.1016/j.eswa.2005.07.006
- Wang, L., & Fan, X. (2004). Missing data in disguise and implications for survey data analysis. *Field Methods*, 16(3), 332–351. doi:10.1177/1525822X03262276
- Wang, L., & Jiang, T. (1994). On the complexity of multiple sequence alignment. *Journal of Computational Biology*, 1(4), 337–348. doi:10.1089/cmb.1994.1.337
- Wang, P. P., Ruan, D., & Kerre, E. E. (2007). *Fuzzy logic: A spectrum of theoretical & practical issues*. Berlin, Germany: Springer.
- Wang, P., & Ji, Q. (2005). *Multi-view face tracking with factorial and switching HMM*. Paper presented at the IEEE Workshop on Applications of Computer Vision (WACV/MOTION05).
- Wang, T., Li, J., Diao, Q., Hu, W., Zhang, Y., & Dulong, C. (2006). *Semantic event detection using conditional random fields*. Paper presented at the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
- Wang, X. (1994). Learning planning operators by observation and practice. In *Proceedings of the International Conference on AI Planning Systems* (pp. 335-340).
- Wang, X., & Peng, G. (2003). Modeling and control for pneumatic manipulator based on dynamic neural network. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* (Vol. 3, pp. 2231-2236).
- Wang, X., Peng, G., & Xue, Y. (2003). Internal model controller with diagonal recurrent neural network for pneumatic robot servo system. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automatics*, Japan (pp. 1064-1069).
- Wang, Y., Klijn, J. G., & Zhang, Y. (2005). Gene expression profiles to predict distant metastasis of lymph-node-negative primary breast cancer. *Lancet*, 365, 67–679.
- Watkins, C. J. C. H. (1989). Learning from delayed rewards. Unpublished doctoral dissertation, Cambridge University.
- Weigend, A. S., & Gerschenfeld, N. A. (1994). *Time series prediction: Forecasting the future and understanding the past*. Reading, MA: Addison-Wesley.
- Weiner, P. (1973). Linear pattern matching algorithm. In *Proceedings of the 14th IEEE Symposium on Switching and Automata Theory* (pp. 1-11).
- Wellner, B., Huyck, M., & Mardis, S. (2007). Rapidly retargetable approaches top de-identification in medical records. *Journal of the American Medical Informatics Association*, 14, 564–573. doi:10.1197/jamia.M2435
- White, H. (1988). economic prediction using neural networks: The case of IBM daily stock returns. In *Proceedings of the Second Annual IEEE Conference on Neural Networks, II* (pp. 451-458).
- Widrow, B., Rumelhart, D. E., & Lehr, M. A. (1994). Neural networks: Applications in industry, business and science. *Communications of the ACM*, 37(3), 93–105. doi:10.1145/175247.175257
- Wiebe, J., & Riloff, E. (2005). *Creating subjective and objective sentence classifiers from unannotated texts*. Paper presented at the Conference on Intelligent Text Processing and Computational Linguistics (CICLING 2005).
- Wiebe, J., & Riloff, E. (2005). *Creating subjective and objective sentence classifiers from unannotated texts*. Paper presented at the Conference on Intelligent Text Processing and Computational Linguistics (CICLING 2005).

- Williams, C. K. I. (2002). On a connection between kernel PCA and metric multidimensional scaling. *Machine Learning*, 46, 11–19. doi:10.1023/A:1012485807823
- Williams, D., Liao, X., Xue, Y., Carin, L., & Krishnapuram, B. (2007). On classification with incomplete data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3), 427–436. doi:10.1109/TPAMI.2007.52
- Williams, R. J., Rumelhart, D. E., & Hinton, G. E. (1986). Learning internal representations by error propagation. In D. E. Rumelhart & J. L. McClelland (Eds.), *Parallel distributed processing: Explorations in the microstructure of cognition* (pp. 318–362). Cambridge, MA: The MIT Press.
- Williamson, J. (1996). Gaussian ARTMAP: A neural network for fast incremental learning of noisy multi-dimensional maps. *Neural Networks*, 9(5), 881–897. doi:10.1016/0893-6080(95)00115-8
- Wilson, R. L., & Sharda, R. (1994). Bankruptcy prediction using neural networks. *Decision Support Systems*, 11, 545–557. doi:10.1016/0167-9236(94)90024-8
- Wilson, T., Wiebe, J., & Hwa, R. (2006). Recognizing strong and weak opinion clauses. *Computational Intelligence*, 22(2), 73–99. doi:10.1111/j.1467-8640.2006.00275.x
- Witte, R., & Bergler, S. (2007). *Fuzzy clustering for topic analysis and summarization of document collections*. Paper presented at the 20th Canadian Conference on Artificial Intelligence.
- Witten, I., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques*. San Francisco: Morgan Kaufmann.
- Wittig, G., & Finnie, G. (1997). Estimating software development effort with connectionist models. *Information and Software Technology*, 39, 469–476. doi:10.1016/S0950-5849(97)00004-9
- Wohlin, C., Petersson, H., Höst, M., & Runeson, P. (2001). Defect content estimation for two reviewers. In *Proceedings of the 12th International Symposium on Software Reliability Engineering* (pp. 340–345).
- Wojcik, R., Hauenstein, L., Sniegoski, C., & Holtry, R. (2007). Obtaining the data. In J. Lombardo & D. Buckeridge (Eds.), *Disease surveillance* (pp. 91–142). New York: Wiley.
- Wolcott, D. L., Marsh, J. T., La Rue, A., Carr, C., & Nissensohn, A. R. (1989). Recombinant human erythropoietin treatment may improve quality of life and cognitive function in chronic hemodialysis patients. *American Journal of Kidney Diseases*, 14, 478–485.
- Wolpert, D., & Macready, W. (2005). Coevolutionary free lunches. *IEEE Transactions on Evolutionary Computation*, 9(6), 721–735. doi:10.1109/TEVC.2005.856205
- Wong, B. K., & Selvi, Y. (1998). Neural network applications in finance: A review and analysis of literature (1990–1996). *Information Management*, 34(3), 129–139. doi:10.1016/S0378-7206(98)00050-0
- Wong, K. Y., Casey, R. G., & Wahl, F. M. (1982). Document analysis system. *IBM Journal of Research and Development*, 26(6), 647–656.
- Wu, G., Chang, E. Y., & Panda, N. (2005). Formulating distance functions via the kernel trick. In *Proceedings of the 11th ACM International Conference on Knowledge Discovery in Data Mining* (pp. 703–709).
- Wu, M.-C., Lin, S.-Y., & Lin, C.-H. (2006). An effective application of decision tree to stock trading. *Expert Systems with Applications*, 31(2), 270–274. doi:10.1016/j.eswa.2005.09.026
- Wu, P., Fang, S., King, R. E., & Nuttle, H. L. W. (1994). Decision surface modelling of apparel retail operations using neural network technology. In *Proc. of the IEEE 1994 Textile, Fiber and Film Industrial Conference*.
- Wu, P., Yang, W., & Wei, N. (2004). An electromagnetism algorithm of neural network analysis – an application to textile retail operation. *Journal of Chinese Institute of Industrial Engineers*, 21(1), 59–67.
- Wu, T., & Brutlag, D. (1995). Identification of protein motifs using conserved amino acid properties and partitioning techniques. In *Proceedings of the 3rd International Conference on Intelligent Systems for Molecular Biology* (pp. 402–410).

Compilation of References

- Wunsch, D. II, Caudell, T., Capps, C., Marks, R., & Falk, R. (1993). An optoelectronic implementation of the adaptive resonance neural network. *IEEE Transactions on Neural Networks*, 4(4), 673–684. doi:10.1109/72.238321
- Wunsch, D., II, & Mulder, S. (2004). Evolutionary algorithms, Markov decision processes, adaptive critic designs, and clustering: Commonalities, hybridization, and performance. In *Proceedings of the IEEE International Conference on Intelligent Sensing and Information Processing*.
- Wunsch, D., II. (1991). *An optoelectronic learning machine: Invention, experimentation, analysis of first hardware implementation of the ARTI neural network*. Unpublished doctoral dissertation, University of Washington.
- Xie, L., Chang, S., Divakaran, A., & Sun, H. (2003). Unsupervised mining of staistical temporal structures in video. In D. D. A. Rosenfeld (Ed.), *Video mining*. Amsterdam: Kluwer Academi Publishers.
- Xie, M. (1991). *Software reliability modelling*. Singapore: World Scientific Publishing.
- Xiong, Z. Y., Zhou, X. S., Tian, Q., Rui, Y., & Huang, T. S. (2006). Semantic retrieval of video - review of research on video retrieval in meetings, movies and broadcast news, and sports. *IEEE Signal Processing Magazine*, 23(2), 18–27. doi:10.1109/MSP.2006.1621445
- Xu, L (2005), Fundamentals, Challenges, and Advances of Statistical Learning for Knowledge Discovery and Problem Solving: A BYY Harmony Perspective, Keynote talk. *Proc. Of Intl. Conf. on Neural Networks and Brain*, Oct. 13-15, 2005, Beijing, China, Vol. 1, 24-55.
- Xu, L. (1995), Bayesian-Kullback coupled YING-YANG machines: unified learning and new results on vector quantization, Proc.ICONIP95, Oct 30-Nov.3, 1995, Beijing, pp977-988. A further version in NIPS8, D.S. Touretzky, et al (Eds.), MIT Press, 444–450.
- Xu, L. (1998). RBF nets, mixture experts, and Bayesian Ying-Yang learning . *Neurocomputing*, 19(1-3), 223–257.
- Xu, L. (2001). Best Harmony, Unified RPCL and Automated Model Selection for Unsupervised and Supervised Learning on Gaussian Mixtures, ME-RBF Models and Three-Layer Nets . *International Journal of Neural Systems*, 11(1), 3–69.
- Xu, L. (2001). BYY harmony learning,independent state space and generalized APT financial analyses . *IEEE Transactions on Neural Networks*, 12(4), 822–849.
- Xu, L. (2002). BYY harmony learning, structural RPCL, and topological self-organizing on unsupervised and supervised mixture models . *Neural Networks*, 15, 1125–1151.
- Xu, L. (2004). Advances on BYY harmony learning: information theoretic perspective, generalized projection geometry, and independent factor auto-determination . *IEEE Transactions on Neural Networks*, 15, 885–902.
- Xu, L. (2004). Temporal BYY encoding, Markovian state spaces, and space dimension determination . *IEEE Transactions on Neural Networks*, 15, 1276–1295.
- Xu, L. (2007), Bayesian Ying Yang Learning, Scholarpedia 2(3):1809, Retrieved from http://scholarpedia.org/article/Bayesian_Ying_Yang_learning.
- Xu, L. (2007b), Rival penalized competitive learning, Scholarpedia 2(8):1810, Retrieved from http://www.scholarpedia.org/article/Rival_penalized_competitive_learning
- Xu, L. (2007), A trend on regularization and model selection in statistical learning: a Bayesian Ying Yang learning perspective, In Duch & Mandziuk (eds.), *Challenges for Computational Intelligence*, Springer-Verlag, 365-406.
- Xu, L. (2007). A unified perspective and new results on RHT computing, mixture based learning, and multi-learner based problem solving . *Pattern Recognition*, 40, 2129–2153.
- Xu, L. (2008), Bayesian Ying Yang System, Best Harmony Learning, and Gaussian Manifold Based Family, In Zurada et al (eds.) Computational Intelligence: Research Frontiers, WCCI2008 Plenary/Invited Lectures, LNCS5050, 48–78.

- Xu, L. (2008). (in press). Machine learning problems from optimization perspective, A special issue for CDGO 07 . *Journal of Global Optimization*.
- Xu, L. (2008), Independent Subspaces, in Ramón, Dopico, Dorado & Pazos (Eds.), *Encyclopedia of Artificial Intelligence*, IGI Global (IGI) publishing company, 903-912.
- Xu, L., Jordan, M. I., & Hinton, G. E. (1994), A Modified Gating Network for the Mixtures of Experts Architecture, Proc. of WCNN94, San Diego, CA, 405-410.
- Xu, L., Jordan, M. I., & Hinton, G. E. (1995), An Alternative Model for Mixtures of Experts, In Tesauro, Touretzky & Leen (eds), *Advances in Neural Information Processing Systems 7*, MIT Press, 633-640.
- Xu, L., Krzyzak, A., & Oja, E. (1993), Rival Penalized Competitive Learning for Clustering Analysis, RBF net and Curve Detection, *IEEE Trans. Neural Networks* 4(4), pp.636-649, An early version on Proc. 1992 IJCNN, Nov.3-6, 1992, Beijing, 665-670.
- Xu, L., Krzyzak, A., & Yuille, A. L. (1994). On Radial Basis Function Nets and Kernel Regression: Statistical Consistency, Convergence Rates and Receptive Field Size . *Neural Networks*, 7(4), 609–628.
- Xu, R., & Wunsch, D., II. (2008). *Clustering*. Hoboken, NJ: Wiley / IEEE Press.
- Xu, Y., Fern, A., & Yoon, S. (2007). Discriminative learning of beam-search heuristics for planning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (pp. 2041-2046).
- Xu, Z., & Khoshgoftaar, T. M. (2004). Identification of fuzzy models cost estimation . *Fuzzy Sets and Systems*, 145, 141–163. doi:10.1016/j.fss.2003.10.008
- Xudong, J., & Ser, W. (2002). Online fingerprint template improvement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(8), 1121–1126. doi:10.1109/TPAMI.2002.1023807
- Yang, Q., Wu, K., & Jiang, Y. (2005). Learning action models from plan examples with incomplete knowledge. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- Yang, Z. R., & Chen, S. (1998). Robust maximum likelihood training of heteroscedastic probabilistic neural networks. *Neural Networks*, 11(4), 739–748. doi:10.1016/S0893-6080(98)00024-0
- Yangarber, R., von Etter, P., & Steinberger, R. (2008). *Content collection and analysis in the domain of epidemiology*. Paper presented at the International Workshop on Describing Medical Web Resources, held in conjunction with the 21st International Congress of the European Federation for Medical Informatics (MIE 2008).
- Yao, X. (1999). Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9), 1423–1447. doi:10.1109/5.784219
- Yarowsky, D., & Florian, R. (2002). Evaluating sense disambiguation across diverse parameter spaces. *Natural Language Engineering*, 8(4), 293–310. doi:10.1017/S135132490200298X
- Yildiz, O., & Alpaydin, E. (2006). Ordering and finding the best of $K>2$ supervised learning algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(3), 392–402. doi:10.1109/TPAMI.2006.61
- Yin, H. (2008). Learning nonlinear principal manifolds by self-organising maps. In A. Gorban, B. Kégl, D. Wunsch, & A. Zinovyev (Eds.), *Principal manifolds for data visualization and dimension reduction* (pp. 68-95). Berlin, Germany: Springer.
- Yin, H., & Allinson, N. (2001). Self-organizing mixture networks for probability density estimation. *IEEE Transactions on Neural Networks*, 12, 405–411. doi:10.1109/72.914534
- Yoon, S. Y., & Lee, S. Y. (1999). Training algorithm with incomplete data for feed-forward neural networks. *Neural Processing Letters*, 10, 171–179. doi:10.1023/A:1018772122605
- Yoon, S., Fern, A., & Givan, R. (2004). Learning domain-specific control knowledge from random walks. In *Proceedings of the International Conference on Automated Planning and Scheduling*.

Compilation of References

- Yoon, S., Fern, A., & Givan, R. (2006). Learning heuristic functions from relaxed plans. In *Proceedings of the International Conference on Automated Planning and Scheduling*.
- Yoon, S., Fern, A., & Givan, R. (2007). Using learned policies in heuristic-search planning. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence* (pp. 1636-1642).
- Yu, D. L., & Chang, T. K. (2005). Adaptation of diagonal recurrent neural network model. *Neural Computing & Applications*, 14, 189–197. doi:10.1007/s00521-004-0453-9
- Yu, L., Wang, S. Y., & Lai, K. K. (2005). A novel non-linear ensemble-forecasting model incorporating GLAR and ANN for foreign exchange rates. *Computers & Operations Research*, 32(10), 2523–2541. doi:10.1016/j.cor.2004.06.024
- Yuille, A. L., & Grzywacz, N. M. (1989). A mathematical analysis of the motion coherence theory . *International Journal of Computer Vision*, 3, 155–175.
- Zadeh, L. (1965). Fuzzy sets. *Information and Control*, 8, 338–353. doi:10.1016/S0019-9958(65)90241-X
- Zadeh, L. A. (1992). The calculus of fuzzy if-then rules. *AI Expert*, 7(3), 23–27.
- Zadeh, L. A. (1994). Soft computing and fuzzy logic. *IEEE Software*, 11(6), 48–56. doi:10.1109/52.329401
- Zadeh, L. A. (1998). Some reflections on soft computing, granular computing and their roles in the conception, design and utilization of information/intelligent systems . *Soft Computing*, 2(1), 23–25. doi:10.1007/s005000050030
- Zadrozny, B., & Elkan, C. (2001). Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers. In *Proceedings of the 8th International Conference on Machine Learning* (pp. 609-616).
- Zadrozny, B., & Elkan, C. (2002). Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 694-699).
- Zayas, I., Converse, H., & Steele, J. (1990). Discrimination of whole from broken corn kernels with image analysis. *Transactions of the American Society of Agricultural Engineers*, 33(5), 1642–1646.
- Zelle, J., & Mooney, R. (1993). Combining FOIL and EBG to speed-up logic programs. In *Proceedings of the International Joint Conference on Artificial Intelligence* (pp. 1106-1113).
- Zemouri, R., Racoceanu, D., & Zerhouni, N. (2003). Recurrent radial basis function network for time series prediction. *Engineering Applications of Artificial Intelligence*, 16, 453–463. doi:10.1016/S0952-1976(03)00063-0
- Zhang, D., Gatica-Perez, D., Bengio, S., & Roy, D. (2005). *Learning influence among interacting Markov chains*. Paper presented at the NIPS.
- Zhang, D., Wai-Kin, K., You, J., & Wong, M. (2003). Online palmpoint identification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9), 1041–1050. doi:10.1109/TPAMI.2003.1227981
- Zhang, G., Patuwo, B. E., & Hu, M. Y. (1998). Forecasting with artificial neural networks: The state of the art. *International Journal of Forecasting*, 14, 35–62. doi:10.1016/S0169-2070(97)00044-7
- Zhang, L., Pan, Y., & Zhang, T. (2004). *Focused named entity recognition using machine learning*. Paper presented at the 27th Annual International ACM SIGIR Conference (SIGIR 2004).
- Zhang, T., Damerau, F., & Johnson, D. (2002). Text chunking based on a generalization of Winnow. *Journal of Machine Learning Research*, 2, 615–637. doi:10.1162/153244302320884560
- Zhang, T., Ramakrishnan, R., & Livny, M. (1996) BIRCH: An efficient data clustering method for very large databases. In *Proceedings of the ACM SIGMOD Conference on Management of Data* (pp. 103-114).

- Zhang, X. S., Huang, J. W., & Roy, R. J. (2002). Modeling for neuromonitoring depth of anesthesia. *Critical Reviews in Biomedical Engineering*, 30(1-3), 131–173. doi:10.1615/CritRevBiomedEng.v30.i123.70
- Zhang, Y., Su, H., Jia, T., & Chu, J. (2005). Rule extraction from trained support vector machines. In *Advances in knowledge discovery and data mining* (LNCS 3518, pp. 61-70). Berlin, Germany: Springer.
- Zheng, Z., & Low, B. T. (1999). Classifying unseen cases with many missing values. In N. Zhong & L. Zhou (Eds.), *Proceedings of the Third Pacific-Asia Conference on Methodologies for Knowledge Discovery and Data Mining*. Beijing, China (LNAI 1574, pp. 370-374). Berlin, Germany: Springer.
- Zipf, G. (1935). *The psycho-biology of language*. Boston: Houghton Mifflin.
- Zitzler, E., Bleuler, S., & Laumanns, M. (2004). A tutorial on evolutionary multiobjective optimization. In X. Gandibleux, et al. (Eds.), *Metaheuristics for multiobjective optimization*. Berlin, Germany: Springer.
- Zitzler, E., Deb, K., & Thiele, L. (2005). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8.
- Zizka, J., Hroza, J., Pouliquen, B., Ignat, C., & Steinberger, R. (2006). *the selection of electronic text documents supported by only positive examples*. Paper presented at the 8es Journees internationales d'Analyse statistique des Donnees Textuelles (JADT 2006).
- Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society*, 67(2), 301–320. doi:10.1111/j.1467-9868.2005.00503.x
- Zucker, J.-D. (1998). Semantic abstraction for concept representation and learning. In R. S. Michalski & L. Saitta (Eds.), *Proceedings of the 4th International Workshop on Multistrategy Learning* (pp. 157-164).
- Zurada, J. M., & Cloete, I. (2005). *Knowledge-based neurocomputing*. Cambridge, MA: MIT Press.

About the Contributors

Emilio Soria received a M.S. degree in Physics in 1992 and the Ph.D. degree in 1997 in electronics engineering from the Universitat de Valencia, Spain. He is an Assistant Professor at the University of Valencia since 1997. His research is centered mainly in the analysis and applications of adaptive and neural systems.

José David Martín-Guerrero was born in Valencia, Spain in 1974. He received a BS degree in Physics (1997), a BS degree in Electronics Engineering (1999), a MS Degree in Electronic Engineering (2001) and a PhD Degree in Electronic Engineering (2004) from the University of Valencia (Spain). He is currently an Assistant Professor at the Department of Electronic Engineering, University of Valencia. His research interests include Machine Learning algorithms and their potential real application. Lately, his research has been especially focused on the study of Reinforcement Learning algorithms.

Rafael Magdalena received the M.S. and the Ph.D. degrees in Physics from the University of Valencia, Spain in 1991 and 2000 respectively. He has also been a lecturer with the Politechnic University of Valencia, a funded researcher with the Research Association in Optics and has held industrial positions with several electromedicine and IT companies. Currently he is a Labour Lecturer in Electronic Engineering with the University of Valencia, position held since 1998. He has conducted research in telemedicine, biomedical engineering and signal processing. He is a Member of the IEICE.

Marcelino Martinez received his B.S. and Ph.D. degrees in Physics in 1992 and 2000 respectively from the Universitat de Valencia(Spain). Since 1994 he has been with the Digital Signal Processing Group at the Department of Electronics Engineering. He is currently an Asistant Professor. He has worked on several industrial projects with private companies (in the areas such as industrial control, real-time signal processing and digital control) and with public funds (in the areas of foetal electrocardiography and ventricular fibrillation). His research interests include real time signal processing, digital control using DSP and biomedical signal processing.

Antonio J. Serrano received a B.S. degree in Physics in 1996, a M.S. degree in Physics in 1998 and a Ph.D. degree in Electronics Engineering in 2002, from the University of Valencia. He is currently an associate professor in the Electronics Engineering Department at the same university. His research interest is machine learning methods for biomedical signal processing.

* * *

Mariano Alcañíz was born in 1962. He completed his M.Sc. in Industrial Engineering at the Technical University of Valencia (UPV) in 1985 and Ph.D. in 1992. Since 1986 he has been a scientific fellow at the Department of Graphical Engineering at the Technical University of Valencia, where he has dealt with Computer Graphics. His current position is Full Professor of Computer Graphics at Technical University of Valencia and director of the Human Centered Technology Lab (LabHuman) composed by a multidisciplinary team of researchers focused on how technology can augment human ability, studying technology-augmented human cognition and action, human computer interaction, and communication applications of emerging technologies.

Nuria Aleixos received a B.Sc. (1992) and Ph.D. degree in Computer Science from the Polytechnic University of Valencia (1999). She worked for private companies developing solutions for commercial CAD systems (1990-1994) and at the Public Research Institute IVIA developing machine vision systems (1995-1996). She joined the Department of Technology at Jaume I University of Castelló (UJI), Spain, in 1996. During 2007 she has been the head of the faculty of Industrial Design at UJI until she joined the Graphics Engineering Department of the Polytechnic University of Valencia in October 2007, where she currently works as associate professor. Her fields of interest are focused on *Modelling methodologies for CAD applications, Computer aided design, Image processing and Calligraphic interfaces*

Kin-fan AU is an associate professor in the Institute of Textiles and Clothing of The Hong Kong Polytechnic University. His research interest is in the business aspects of fashion and textiles, particularly in the area of global trading of fashion and textile products. Dr. Au has published many papers in textiles and related journals on topics of world trading, offshore production and modelling of textiles and apparel trade.

Paulo J. Azevedo received is MSc and PhD degrees in Computing from Imperial College at the University of London in 1991 and 1995. He is an Auxiliar Professor at Departamento de Informática of the University of Minho. His research interests include bioinformatics, data mining, machine learning, data warehousing and logic programming.

Dr. Raju S. Bapi obtained BTech (EE) from Osmania University, India, MS and PhD from University of Texas at Arlington, USA. He has over 10 years of teaching and research experience in neural networks, machine learning and artificial intelligence andtheir applications. Currently he is a Reader in the Department of Computer and Information Sciences, University of Hyderabad. He has over 50 publications (Journal/Conference). His main research interests include Biological and Artificial NeuralNetworks, Neural and Cognitive Modelling, Machine Learning, Pattern Recognition, Neuroimaging and Bioinformatics.

Teresa M.A. Basile got the Laurea degree in Computer Science at the University of Bari, Italy (2001). In March 2005 she obtained the Ph.D. at the University of Bari defending a dissertation titled “A Multistrategy Framework for First-Order Rules Learning.” Since April 2005, she is a research at the Computer Science Department of the University of Bari. Her research interests concern the investigation of symbolic machine learning techniques. She is author of about 60 papers published on National and International journals and conferences/workshops proceedings and was/is involved in various National and European projects.

About the Contributors

Antonio Bella finished his degree in Computer Science at the Technical University of Valencia in 2004 and started Ph.D. studies in Machine Learning in the Department of Information System and Computation at the same university. He also obtained a MSc in Corporative Networks and Systems Integration (2005) and he is currently completing a degree in Statistical Science and Technology at the University of Valencia.

Marenglen Biba is a third year Ph.D student at the Department of Computer Science, University of Bari, Italy. He received his Masters Degree in Computer Science (summa cum laude) from the same department in 2004. His main research area is Machine Learning with a particular interest in the integration of logical and statistical learning with applications in computational biology, social and biological networks analysis and document understanding.

José Blasco received a B.Sc. (1994) and Ph.D. degree in Computer Science from Polytechnic University of Valencia, UPV, (2001). He also received a M.Sc. in Computer Aided Design and Manufacturing (CAD/CAM) from UPV in 1995. He worked during two years at Industrial Business Machine (IBM) Spain as system analyst before he joined at the Public Research Institute, IVIA in 1996. Since then, he is the manager of the Computer Vision Lab at the Agricultural Engineering Centre of IVIA. His field of interest is mainly focused on *Computer Vision, Image Processing, Real time inspection and Machine vision*.

Dr. Mahil Carr currently teaches at the IDRBT at Hyderabad. Dr. Mahil Carr has a Bachelor's degree in Mathematics from the American College, Madurai Kamaraj University, Madurai, a Master of Computer Applications from St. Joseph's College, Bharathidasan University, Trichy and was awarded a doctoral degree in Information Systems from the City University of Hong Kong. His current research interests are in the areas of software engineering, information systems security and electronic/mobile commerce. Dr. Carr is on the editorial board of the International Journal of E-Services and Mobile Applications (IJESMA) and the International Journal of Information Systems and Social Change (IJISSC).

Manuel Castejón-Limas received his engineering degree from the Universidad de Oviedo in 1999 and his Ph.D. degree from the Universidad de La Rioja in 2004. From 2002 he teaches project management at the Universidad de León. His research is oriented towards the development of data analysis procedures that may aid project managers on their decision making processes.

Tsan-Ming Choi is an associate professor at the Hong Kong Polytechnic University. His current research interests mainly focus on supply chain management. He has published in journals such as Computers and Operations Research, Decision Support Systems, European Journal of Operational Research, IEEE Transactions (various), International Journal of Production Economics, Journal of Industrial and Management Optimization, Journal of the Operational Research Society, Omega: The International Journal of Management Science, etc. He is a member of IEEE and INFORMS.

Sanjoy Das received his Ph.D. in Electrical Engineering from Louisiana State University, in 1994. He received postdoctoral training at the University of California, Berkeley between 1994 and 1997, and worked for three years as a Senior Scientist at ITT Systems & Sciences, Colorado Springs. At present he is an Associate Professor in the Department of Electrical & Computer Engineering, at Kansas State

University, where he has been employed since 2001. Prof. Das has co-authored over 130 research papers, and has supervised several masters and doctoral students. His areas of interest are evolutionary algorithms, neural networks, multi-objective optimization, and artificial life.

Tomás de la Rosa, is an assistant lecturer at the University Carlos III de Madrid. He belongs to the Planning and Learning research group and his research topics are heuristic planning and Case-Based Reasoning.

Yi Ding received the B.S. degree in communication engineering from Xi'an University of Technology, China, and the M.S. degree in communication engineering from Xidian University, China, in 2002 and 2005 respectively. He is currently working toward the Ph.D. degree in the School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK. His research interests include multimedia content analysis, pattern recognition, image processing, and computer vision.

Jie Du is a Ph.D. student in the Department of Information Systems at the University of Maryland, Baltimore County. She received her M.S. in Information Systems from Beijing Jiaotong University in 2004 and her B.A. in Information Management and Information Systems from Southwest Jiaotong University in 2001. Her research interests are intelligent financial investing.

Floriana Esposito, ECCAI Fellow, is since 1994 Full Professor of Computer Science at the University of Bari. Among her scientific interests there are similarity based learning, integration of numerical and symbolic methods in learning and classification, logical foundations of conceptual learning techniques, revision of logical theories by incremental learning, multistrategy learning, inductive logic programming. The major application fields are Document processing and Digital Libraries and, recently, Bioinformatics. She is author of more than 250 papers which published in international journals and Proceedings. She is in the pc of the most important scientific conferences on Machine Learning and Artificial Intelligence.

Guoliang Fan received his B.S. degree in Automation Engineering from Xi'an University of Technology, Xi'an, China, M.S. degree in Computer Engineering from Xidian University, Xi'an, China, and Ph.D. degree in Electrical Engineering from the University of Delaware, Newark, DE, USA, in 1993, 1996, and 2001, respectively. Since 2001, Dr. Fan has been with the School of Electrical and Computer Engineering, Oklahoma State University (OSU), Stillwater, OK, where he is currently an Associate Professor. Dr. Fan is a recipient of the National Science Foundation (NSF) CAREER award (2004). He received the Halliburton Excellent Young Teacher Award (2004), Halliburton Outstanding Young Faculty award (2006) and IEEE-OSU Outstanding Faculty Award (2008). His research interests are image processing, computer vision, and machine learning.

Mohammad Abdul Haque Farquad is holding Masters Degree in Information Systems from Osmania University and currently pursuing Ph. D in Computer Science from Department of Computer and Information Sciences, University of Hyderabad and Institute for Development and Research in Banking Technology (IDRBT), Hyderabad. His research interests include Machine Learning, Soft Computing, Artificial Neural Network and Data Mining. He is financially supported by IDRBT.

About the Contributors

Stefano Ferilli was born in 1972. He got a PhD in Computer Science at the University of Bari, and is currently an Associate Professor at the Computer Science Department of the University of Bari. His research interests include Logic and Algebraic Foundations of Machine Learning, Inductive Logic Programming, Multistrategy Learning, Knowledge Representation, Digital Document Processing and Digital Libraries. He participated in various National and European (ESPRIT e IST) projects on these topics, and is a (co-)author of more than 100 papers published on journals, books and proceedings of national and international workshops/conferences.

Carlos Fernández was born in Madrid, Spain (1963). He received the Ph.D degree in Agricultural Engineering from the Politechnic University at Madrid, Spain, in 1993. He made a post doctorate stage at University of California (Animal Science Department), Davis during 1993 and 1994. He is Associate Professor at the Politechnic University of Valencia. His research is centered in data analysis, modelling and simulation in Animal Science. He specially worked in animal nutrition and production systems for small ruminant animals. He has worked on several industrial projects with private companies (in the areas natural aditives for animal nutrition) and with public funds (in the areas of nutrition feeding, sustentability and animal performance).

Susana Fernández Arregui, is an assistant lecturer at the University Carlos III de Madrid. She received her PhD in learning search control for classical planning in 2006. She belongs to the Planning and Learning research group. Her research topics are learning search control for planning and the study of the role of emotions in AI.

Pedro Gabriel Ferreira has a degree in Systems and Informatics Engineer and a PhD in Artificial Intelligence from University of Minho in 2002 and 2007. He is now a Bioinformatics Pos-Doctoral Researcher at Center for Genomic Regulation, Barcelona, Spain. His research interests include Data Mining and Machine Learning and its applications to Computational Biology and Bioinformatics

Cèsar Ferri is an associate professor of computer science at the Department of Information Systems and Computation, Technical University of Valencia, Spain, where he has been working since 1999. He obtained his BSc at the Technical University of Valencia, and his MSc at the University of Pisa, Italy. His research interests include machine learning, cost-sensitive learning, relational data mining, and declarative programming. He has published several journal articles, books, book chapters and conference papers on these topics.

Aníbal R. Figueiras-Vidal received the Ph.D. degree from Universidad Politécnica de Barcelona in 1976. He is currently a Professor in Signal Theory and Communications at Universidad Carlos III de Madrid. His research interests include digital signal processing, digital communications, neural networks, and learning theory; and he has either authored or coauthored more than 300 journal and conference papers in these areas. Dr. Figueiras-Vidal received a Honoris Causa Doctorate from the Universidad de Vigo in 1999. He is the President of the Spain Royal Academy of Engineering.

Patrick Flandrin is currently with the Physics Department at Ecole Normale Supérieure de Lyon, where he works as a CNRS senior researcher. His research interests include mainly nonstationary signal processing at large (with emphasis on time-frequency and time-scale methods), scaling processes and

complex systems. He was awarded the Philip Morris Scientific Prize in Mathematics in 1991, the SPIE Wavelet Pioneer Award in 2001 and the Michel Monpetit Prize from the French Academy of Sciences in 2001. He is a Fellow of the IEEE since 2002.

Todor D. Ganchev received the Diploma Engineer degree in Electrical Engineering from the Technical University of Varna, Varna, Bulgaria, in 1993 and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Patras, Patras, Greece, in 2005. From February 1994 to August 2000, he consequently held Engineering, Research, and Teaching Staff positions at the Technical University of Varna. Since September 2000, he has been a Research Fellow at the Wire Communications Laboratory, University of Patras, Greece. Currently, he holds a Senior Researcher position at the Wire Communications Laboratory. His research interests are in the areas of pattern recognition, signal processing, and applications.

Pedro J. García-Laencina received the MS degree in telecommunication engineering in 2004 from Universidad Politécnica de Cartagena, Spain. He is currently a researcher pursuing his Ph. D. degree at the same university. His research interests are machine learning, signal processing, statistical pattern recognition, and image processing. He is a student member of the IEEE.

Adam E. Gaweda received the M.Eng. degree in Electrical Engineering from Czestochowa University of Technology, Poland, and the Ph.D. degree in Computer Science and Engineering from University of Louisville, Louisville, KY, in 1997 and 2002, respectively. In 2002 he joined the Department of Medicine, Division of Nephrology, University of Louisville, where he currently holds the position of Assistant Professor. His research interests focus on application of computational intelligence and control methods to pharmacodynamic modeling and adaptive treatment planning.

Juan Gómez-Sanchis received a B.Sc. degree in Physics (2000) and a B.Sc. degree in Electronics Engineering from the University of Valencia (2003). He worked during one year at professor in the Department of Electronics Engineering at the Universitat de València in pattern recognition using neural networks. He joined at the Public Research Institute IVIA in 2004, developing is Ph.D. in hyperspectral computer vision systems applied to the agriculture.

Ana González-Marcos received her engineering and Ph.D. degrees from the Universidad de La Rioja in 2006. She taught project management at the Universidad de León from 2003 to 2007. In 2007 she moved to the Universidad de La Rioja, where she works as a Lecturer in the Department of Mechanical Engineering. Her research interests include the application of multivariate analysis and artificial intelligence techniques in order to improve the quality of industrial processes. Research projects usually concerned real processes, such as steel making and automotive industries.

Prof. Dr. Alexander Gorban obtained his PhD degree in differential equations and mathematical physics in 1980, and Dr. Sc. degree in biophysics in 1990. He holds now a chair of Applied Mathematics at the University of Leicester, UK, and he is the Chief Scientist at the Institute for Computational Modelling Russian Academy of Sciences (Krasnoyarsk, Russia). His scientific interest include interdisciplinary problem of model reduction, topological dynamics, physical and chemical kinetics, mathematical biology and data mining.

About the Contributors

Juan F. Guerrero received the B.Sc. degree in physics and the Ph.D. degree in electronic engineering in 1985 and 1988, respectively, from the Universitat de Valencia, Spain. Since 1985, he has been with the Digital Signal Processing Group (GPDS), Department of Electronic Engineering, University of Valencia, where he is an Associate Professor. His research interests include biomedical digital signal processing and biosignal instrumentation.

José Hernández-Orallo, BSc, MSc (Computer Science, Technical University of Valencia, Spain), MSc (Computer Science, ENSEA, Paris), Ph.D. (Logic, University of Valencia). Since 1996, he has been with the Department of Information Systems and Computation, Technical University of Valencia, where he is currently an Associate Professor. His research interests centre on the areas of artificial intelligence, machine learning, data mining, data warehousing and software engineering, with several books, book chapters, journal and conference articles on these topics.

Paul Honeine received the Dipl.-Ing. degree in mechanical engineering in 2002 and the M.Sc. degree in industrial control in 2003, both from the Faculty of Engineering, the Lebanese University, Lebanon. In 2007, he received the Ph.D. degree in System Optimisation and Security from the University of Technology of Troyes, France, and was a Postdoctoral Research associate with the Systems Modeling and Dependability Laboratory, from 2007 to 2008. Since September 2008, he has been an assistant Professor at the University of Technology of Troyes, France. His research interests include nonstationary signal analysis, nonlinear adaptive filtering, machine learning, and wireless sensor networks.

Sergio Jiménez Celorio, is an assistant lecturer at the University Carlos III de Madrid. He belongs to the Planning and Learning research group and his research topics are relational learning and planning under uncertainty.

Chin Kim On was born in Tawau, Sabah, Malaysia, in 1979. The author received his Master Degree in Software Engineering with Universiti Malaysia Sabah in 2005. The author's research interests included the evolutionary multi-objective optimization, evolutionary robotics, neural networks, and biometric security systems with the main focus on fingerprint and voice recognition. He is currently pursuing his PhD in the area of evolutionary robotics using multi-objective optimization for RF-localization and swarming behaviors with Universiti Malaysia Sabah. Currently, he has over 15 publications. Mr. Kim On is a student member of IEEE and IAENG societies.

Jorge Larrey-Ruiz received the M.S. degree in Telecommunications Engineering in 2000 from the Universidad Politécnica de Valencia, Spain, and the Ph.D. in the same field in 2008 from the Universidad Politécnica de Cartagena, Spain, where he is currently a researcher and Assistant Professor of Signal and Communications Theory. His technical interests include image registration and fusion, and digital filtering.

Dapeng Li received his B. Eng. from Xi'an University of Technology, China, and M. Eng. from Shanghai University, China, in 2001 and 2004 respectively. Between 2004 and 2005, he worked as a full-time research assistant in the Department of Electrical Engineering at Hong Kong Polytechnic University. Between 2005 and 2006, he worked as a power system engineer in the Hygrand Electronic Equipment Company, China. At present he is pursuing a Ph.D. in under the guidance of Prof. Das and

Prof. Pahwa at Kansas State University, in the applications of multi-objective evolutionary algorithms to power distribution systems.

Francisco J. Martínez-de-Pisón graduated in Industrial Engineering at the Universidad de La Rioja in 1999. In 2003 he completed a PhD degree at the same university in the area of industrial optimisation through data mining techniques. His main research interests are data mining; pattern recognition; artificial intelligence and industrial optimisation. He has published several papers in journals and is author of several books and industrial patents. He has been researcher of several Spanish and European data mining Projects. Currently, he supervises several MSc and PhD students in the area of data mining.

Nicola Di Mauro got the Laurea degree in Computer Science at the University of Bari, Italy. In March 2005 he discussed a Ph.D. thesis in Computer Science at the University of Bari titled “First Order Incremental Theory Refinement” facing the problem of Incremental Learning in ILP. Since January 2005, he is an assistant professor at the Department of Computer Science, University of Bari. His research activities concern Relational Learning. He is author of about 60 papers published on international and national journals and conference proceedings. He took part to European and national projects.

Enrique Moltó received his B.Sc. degree (1987) and Ph.D. degree in Agricultural Engineering from the Polytechnic University of Valencia (1991). He also received a B.Sc. degree in Computer Science at Polytechnic University of Valencia (1991). Since 1987 he worked as engineer at Polytechnic University of Valencia, until he joined IVIA in 1994, where he is currently head of the Agricultural Engineering Centre. He has chaired the fields of interest “Machine Vision” and “Automation and Emerging Technologies” of the European Society of Agricultural Engineers (EurAgEng) (1994-2004) and has been founder and member of the governing board of the equivalent national society (SEA). His research is focused on *computer vision, real time inspection, electronic sensors and agricultural engineering*

Carlos Monserrat-Aranda is a Computer Engineer of the LabHuman research group of the Technical University of Valencia. He is a Computer Engineer and has obtained his Doctoral studies in 1999 obtaining the highest qualification. He is currently an associated professor at the Computer Science Department of the Technical University of Valencia. His research interests include real-time deformable models, virtual reality and simulation of physical behavior in virtual environments.

Juan Morales-Sánchez received the B.S. degree in 1992 from the Universidad de Alcalá, Spain, the M.S. degree in Telecommunications Engineering in 1996 from the Universidad Politécnica de Valencia, Spain, and the Ph.D. in the same field in 2005 from the Universidad Politécnica de Cartagena (UPCT), Spain. Since 1999, he has been involved in different research and development projects at UPCT, where he is currently Assistant Professor of Signal and Communications Theory. His technical interests include image processing and medical imaging.

Dr. Iván Olier completed his PhD at Technical University of Catalonia (UPC), Barcelona, Spain, in 2008. He is currently a postdoctoral researcher at Autonomous University of Barcelona (UAB), Spain. Research interests include machine learning, time series modeling, as well as their application in biomedicine.

About the Contributors

Joaquín Ordieres-Meré is a full professor at Universidad Politécnica de Madrid, academically responsible for project management topics and with research interests close related to knowledge management and knowledge discovering from data. Usually his team implement and develop new algorithms in order to address in a better way problems under consideration. He has special knowledge and interest in knowledge management in industrial applications, including refinement of models or even, development of new models for processes like steel production (flat and section products), extrusion of rubber for automotive components as well as environmental applications from waste management plants.

Frank Padberg received the master's degree in mathematics from the University of Erlangen, Germany, and the PhD degree in computer science from Saarland University, Germany. He works on fundamental topics in software engineering, including software process optimization, software reliability, and lean software development. Before joining the university, he gave seminars in the industry on networking technology and operating systems. He has been supported by fellowships and research grants from the Deutsche Forschungsgemeinschaft DFG. Recently, he was ranked among the Top 50 International Scholars in Software Engineering. Currently, he holds a prestigious Heisenberg research fellowship awarded by the DFG.

B. K. Panigrahi has completed his B. Sc. Engg, ME and Ph.D in 1990, 1995 and 2004 respectively. He is involved in teaching since 1990 and at present works as Assistant Professor in the Department of Electrical Engineering, IIT, Delhi, India. He is a senior member of IEEE. Prof. Panigrahi has published nearly 75 papers in international journals and international conference proceedings of repute. His current research area includes development of soft computing and evolutionary computing techniques and their engineering applications. In particular his research focuses in the application of the above techniques to power system planning, operation and control.

S. S. Pattnaik is presently working as Professor and Head of Electronics and Communication Engineering and Educational Television Centre of the National Institute of Technical Teachers' Training and Research (NITTTR), Chandigarh, India. Prof. Pattnaik received his Ph.D from Sambalpur University, India, and postdoctoral training from the University of Utah, USA. He has nearly 20 years of teaching and research experience. During this period he has published more than 155 research papers. Prof. Pattnaik is senior member of IEEE, as well as several national organization. His areas of interest are soft computing and application to antenna, video, image and MIMO systems.

Eduard Plett received his Master's degree in Electrical Engineering from Kansas State University in 2003. At present, he is an Assistant Professor in the department of Electronic & Computer Engineering Technology at Kansas State University, Salina. He is also working on his PhD in Electrical Engineering. His areas of interest are control systems and artificial intelligence. He also worked as an electronic technician for 12 years and as an electrical engineer for 5 years. Most recently he has been working as a consultant for the Advanced Manufacturing Institute in Manhattan, KS, designing and building control systems for automated testing machines

Roy Rada is a professor in the Department of Information Systems at the University of Maryland, Baltimore County. He received his Ph.D. in computer science from University of Illinois at Champaign-Urbana in 1980. His research interest is evolutionary computation applied to finance.

María José Ramírez-Quintana received the BSc from the University of Valencia (Spain) and the Msc and PhD in Computer Science from the Technical University of Valencia (Spain). She is currently an associate professor at the Department of Information Systems and Computation, Technical University of Valencia. She has given several courses on software engineering, functional and logic programming, and multiparadigm programming. Her research interests include multiparadigm programming, machine learning, data mining algorithms, model combination and evaluation, and learning from structured data, with more than 60 publications in these areas, including journal articles, books, book chapters and conference contributions.

Dr. Vadlamani Ravi is an Assistant Professor in IDRBT, Hyderabad since April 2005. He holds a Ph.D. in Soft Computing from Osmania University, Hyderabad & RWTH Aachen Germany. Earlier, he was a Faculty at NUS, Singapore for three years. He published 62 papers in refereed Journals / Conferences and invited book chapters. He edited “*Advances in Banking Technology and Management: Impacts of ICT and CRM*” published by IGI Global, USA. He is a referee for several international journals and on the Editorial board of IJIDS, IJDATS, IJISSS, IJSDS & IJITPM. He is listed in the Marquis Who’s Who in the World in 2009.

Cédric Richard received the M.S. degree in 1994 and the Ph.D. degree in 1998 from Compiègne University of Technology, France, in electrical and computer engineering. Since 2003, he is a Professor at the Systems Modelling and Dependability Laboratory, Troyes University of Technology. His current research interests include statistical signal processing and machine learning. Dr. Richard is the author of over 100 papers. He is a Senior Member of the IEEE, and serves as an associate editor of the IEEE Transactions on SP since 2007. He is a member of the SPTM technical committee of the IEEE SP society since 2009.

María José Rupérez was born in Murcia, Spain in 1973. She received a B.Sc. Degree in Physics by the University of Valencia in Spain (2001). She is a research fellow in LabHuman group from 2005. She is working towards the M.Sc Degree in non-linear elasticity into the PhD program of Mechanics and Material Engineering of the Technical University of Valencia in Spain. Her main areas of research are Finite Elements Methods and Deformable Models, which have been mainly applied to the study of the shoe upper deformation in gait.

Albert Ali Salah worked as a research assistant in the Perceptual Intelligence Laboratory of Boğaziçi University in Istanbul, where he was part of the team working on machine learning, pattern recognition and human-computer interaction. After receiving his PhD in Computer Engineering, he joined the Signals and Images research group at the Center for Mathematics and Computer Science (CWI) in Amsterdam. With his work on face biometrics, he received the inaugural European Biometrics Research Award of the European Biometrics Forum (EBF) in 2006. His recent scientific assignments related to biometrics include program committee memberships for *European Workshop on Biometrics and Identity Management* (BIOID’08), biometrics track of *Int. Conf. of Pattern Recognition* (ICPR’08), and *Int. Conf. on Biometrics* (ICB’09).

José-Luis Sancho-Gómez received the Ph.D. degree in telecommunication engineering from Universidad Carlos III de Madrid in 1999. He is currently an Associate Professor of the Universidad

About the Contributors

Politécnica de Cartagena. His research areas include digital signal processing, digital image processing, statistical pattern recognition and machine learning.

Azali Saudi was born in Sabah, Malaysia in 1974. The author received his Master in Artificial Intelligence from the University of Edinburgh at the Department of Artificial Intelligence in 2001, researching Mobile Robotics that relies upon behavior-based architectures. The author's main research interest is in the area of mobile robotics, robot vision, and software engineering. Since 2001, he has been a Lecturer in computer science at the School of Engineering and Information Technology, Universiti Malaysia Sabah. Currently, he has over 30 publications in the areas of artificial intelligence, particularly robotics-related work and software development.

Jude Shavlik is a Professor of Computer Sciences and of Biostatistics and Medical Informatics at the University of Wisconsin, and is a Fellow of the Association for the Advancement of Artificial Intelligence. He served for three years as editor-in-chief of the AI Magazine, serves on the editorial board of about a dozen journals, and has served as chair or co-chair of several conferences including ICMB, ICML, KCAP, ICDM, and ILP. His current research interests include machine learning and computational biology, with an emphasis on using rich sources of training information, such as human-provided advice.

Marina Sokolova, Ph.D.: Marina Sokolova's current interests include applications of Machine Learning to text analysis, text anonymization and privacy protection in free-form unstructured texts, and text data mining of medical texts and health care records. She has worked in Machine Learning (data-based algorithms, statistical learning theory, measures for performance evaluation, learning from multi-modal data) and in application of Machine Learning to NLP (opinion mining, analysis of human communication, electronic negotiations).

Zhan-Li Sun received his PhD degree from the University of Science & Technology of China in 2005. He worked as a Research Associate at The Hong Kong Polytechnic University in 2006–2007 and a Research Fellow at Nanyang Technological University in 2007–2008. He is currently with the National University of Singapore. His research interests include machine learning, signal and image processing.

Stan Szpakowicz, Ph.D.: Stan Szpakowicz's current interests include text summarization (including summarization of literary prose), automatic and semi-automatic development of lexical resources (in particular Roget's Thesaurus and Polish WordNet), sentiment analysis and applications of Machine Learning in NLP. He has worked in NLP (for nearly 40 years) on grammars, parsers, user interfaces, semantic relations and semantic relatedness, language of negotiations, text summarization and emotion analysis.

Jason Teo received his Doctor in Information Technology from the University of New South Wales @ ADFA in 2003, researching Pareto artificial evolution of virtual legged organisms. His current research interest is in the application of evolutionary multi-objective optimization to robotics, games, and co-evolution. He is currently Deputy Dean and Senior Lecturer at the School of Engineering and Information Technology, Universiti Malaysia Sabah. He has over 100 publications in the areas of evolutionary

computing, artificial life, evolutionary robotics and swarm intelligence. He has been a professional member of the ACM since 2005 and IEEE since 2006.

Lisa Torrey is a 2009 doctoral candidate in Computer Science at the University of Wisconsin, working under the supervision of Professor Jude Shavlik in the area of machine learning. Her dissertation research is on transfer learning, and in particular, on the transfer of relational knowledge in the context of reinforcement learning. In fall 2009, she will become an Assistant Professor in the department of Mathematics, Computer Science, and Statistics at St. Lawrence University.

Dr. Alfredo Vellido received his degree in Physics from the Department of Electronics and Automatic Control of the University of the Basque Country (Spain) in 1996. He completed his PhD at Liverpool John Moores University (UK) in 2000. After a few years of experience in the private sector, he briefly joined Liverpool John Moores University again as research officer in a project in the field of computational neurosciences. He is a Senior Researcher at Technical University of Catalonia. Research interests include, but are not limited to, pattern recognition, machine learning and data mining, as well as their application in medicine, market analysis, ecology, and e-learning, on which subjects he has published widely.

Rafael Verdú-Monedero received the MS degree in telecommunications engineering in 2000 from the Universidad Politécnica de Valencia (UPV), and the PhD in the same field in 2005 from the Universidad Politécnica de Cartagena (UPCT). He is currently a researcher and assistant teacher in signal and communications theory at UPCT. His technical interests are modeling and inverse problems in image analysis. He is an associate member of the IEEE since 1999.

K. Vinaykumar obtained Bachelor's degree of Technology in Computer Science from Jawaharlal Nehru Technological University, Hyderabad. He holds M. Tech (Information Technology) with specialization Banking Technology & Information Security from Institute for Development and Research in Banking Technology and University of Hyderabad, Hyderabad. Presently he is working as Software Engineer in ROCSYS Technologies, Hyderabad. His research interests include Software Engineering, Neural Networks, Advanced Algorithms and Data Structures.

Donald Wunsch, PhD, is the M.K. Finley Missouri Distinguished Professor at Missouri University of Science & Technology. Previously, he was with Texas Tech, Boeing, Rockwell International, and International Laser Systems. Dr. Wunsch has over 275 publications, has attracted over \$5.9M in research funding, and produced thirteen PhD graduates in three disciplines. His key contributions are in adaptive resonance and reinforcement learning hardware and applications, neurofuzzy regression, improved Traveling Salesman Problem heuristics, clustering, and bioinformatics. He is an IEEE Fellow, 2005 International Neural Networks Society (INNS) President, and Senior Fellow of the INNS.

Lei Xu, IEEE Fellow, Fellow of International Association for Pattern Recognition, and member of European Academy of Sciences. Received PhD from Tsinghua Univ in 1997, he worked at several universities, including Peking Univ, Harvard and MIT. He joined Chinese Univ Hong Kong in 1993 and became chair professor since 2002. He served as associate editor for several journals, governor of international neural network society(INNS), past president of Asian-Pacific Neural Networks

About the Contributors

Assembly(APNNA), and Fellow committee member of IEEE Computational Intelligence Society. He has received 1993 Chinese National Nature Science Award, 1995 INNS Leadership Award and 2006 APNNA Outstanding Achievement Award.

Rui Xu, PhD, is a research associate in the Department of Electrical and Computer Engineering at Missouri University of Science & Technology. His main research interests include computational intelligence, machine learning, data mining, neural networks, pattern classification, clustering, and bioinformatics. Dr. Xu is a member of the IEEE, the IEEE Computational Intelligence Society (CIS), and Sigma Xi.

Yong Yu is currently a postdoctoral research associate at the Hong Kong Polytechnic University. He received his PhD from the Hong Kong Polytechnic University. His current research interest is in artificial intelligence, forecasting, decision support systems.

Dr. Andrei Zinovyev obtained his university formation in theoretical physics (cosmology). In 2001 he obtained his PhD degree in Computer Science at the Institute for Computational Modelling of Russian Academy of Sciences (Krasnoyarsk, Russia). After defending his PhD, he moved to France, at Institut des Hautes Etudes Scientifiques in Bures-sur-Yvette to work as a post-doc during 3 years in the mathematical biology group of Professor Misha Gromov. Since January 2005, he is the head of the computational systems biology of cancer team at Institut Curie (INSERM Unit U900 “Bioinformatics and Computational Systems Biology of Cancer”) in Paris. His main area of scientific expertise is bioinformatics, systems biology of cancer, dimension reduction in high-throughput data analysis and model reduction in the dynamical models

Index

A

adaptive dynamic back-propagation (ADBP)
algorithm 205
adaptive network-based fuzzy inference systems (ANFIS) 406, 413, 414, 415, 416, 417, 418, 419, 420, 421, 426
adaptive resonance theory 6, 26
alternative ME (AME) 62, 92
anemia 265, 266, 267, 269, 270, 273, 274, 275, 276
ANN xvii, xviii, xxi
anomaly detection 109, 110, 111, 112, 113, 115, 116, 124, 125, 126, 127
artificial immune system (AIS) 109, 110, 126
artificial intelligence 371, 374
artificial neural network (ANN) 387, 388, 389, 390, 391, 392, 396, 397, 399, 403, 429, 439, 575, 576, 579, 582, 585, 586, 588, 598
artificial neuron 439
asset valuation 376, 377, 378, 379, 380, 382
automatic model selection 60, 66, 67, 74, 75, 76, 78, 81, 93
autonomous mobile robots 585, 588, 597, 598
autoregressive feedback 197, 200

B

Bayesian estimation 5
Bayesian information criterion (BIC) 390, 391, 393, 397, 403
Bayesian learning 537, 538
Bayesian model 246
Bayesian techniques 495, 498

Bayesian transfer 246, 247, 248
Bayesian Ying Yang learning 73, 74, 85, 86, 91, 93
Bayesian Ying-Yang system 93
Bayes theorem 483, 498
biologically inspired algorithm 127
biometric 539, 540, 541, 542, 543, 544, 545, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559, 560
biometric fusion 542, 552, 554, 559, 560
biometric system 544, 551, 560
biometric template 539, 543, 544, 545, 551, 560
BIRCH algorithm 4, 26
Boolean classification tasks 246
bootstrap effect 619
bounded-input bounded-output (BIBO) 205

C

calibration measure 131, 132, 133, 134, 146
calibration technique 129, 146
cascaded extensions 92
centroids 180, 181, 182, 186
class calibration 130, 146
classification and regression trees (CART) 502, 506, 509, 510, 513, 514, 515, 517
closed loop control system 439
closed world assumption 600, 619
cluster analysis 1, 2, 3, 4, 21
clustering 176, 177, 191, 193, 194, 352, 362, 363, 367, 368, 369, 371, 372, 374
clustering algorithms 1, 3, 4, 21, 22, 26
computational learning theory 376, 384

concept language 619
conditional random fields (CRF)
 338, 339, 340, 341
confusion matrix 136, 146
corpus 304, 309, 310, 311, 312, 314, 315,
 316, 317, 321, 322, 323
CURE algorithm 4, 24

D

dairy goat 456
data visualization 177, 179, 194
decision tree (DT) 307, 309, 310, 311, 315,
 317, 328, 333, 338, 340, 341, 404,
 406, 413, 415, 416, 417, 420, 421,
 426, 429, 435, 439
decision tree learning 376
defect content 520, 521, 522, 523, 526,
 527, 528, 531, 532, 535, 536, 537,
 538
deflation 33
detector 110, 111, 112, 113, 114, 115, 116,
 117, 118, 119, 120, 123, 124, 127
diagonal recurrent neural network (DRNN)
 202, 203, 204, 205, 212, 220
differential evolution (DE) algorithm
 577, 580, 598
digital libraries 374
distributional accuracy 168, 173
distribution calibration in regression 130, 146
document clustering 4, 19, 24, 26, 27
document image understanding 368, 369, 374
document processing 348, 349, 350, 351,
 353, 370, 374
dynamic evolving neuro-fuzzy inference sys-
tem (DENFIS) 501, 502, 506, 507,
 510, 511, 512, 513, 514, 516, 518
dynamic neural networks 196, 222

E

elasticity test 572
electromyography (EMG) 565, 573
elitism 99, 108
EM algorithm 464, 466, 468, 469, 470,
 471, 473, 474, 475, 478, 480
empty space phenomenon 526, 538
end stage renal disease 276

ensemble forecasting method 518
entity recognition 302, 322, 323
entropic prior 460, 473, 474, 477, 478, 480
erythropoiesis 276
erythropoietin 265, 266, 274, 276
Euclidean distance 13, 14
evolutionary algorithm 114, 115, 127
evolutionary computation (EC) 375, 376,
 378, 379, 380, 383, 386, 389, 390,
 392, 403, 575, 598
evolutionary multi-objective optimization
(EMO) 574, 575, 576, 577, 578,
 579, 581, 582, 585, 586, 588, 593,
 598
evolutionary neural network (ENN) xvii,
 387, 388, 390, 393, 394, 396, 397,
 399, 403
evolutionary robotics
 574, 575, 596, 597, 598
expanded range approximation (ERA) algo-
rithm 567, 573
expectation-maximization (EM) algorithm
 177, 180, 182, 183
expert systems 375, 379, 380, 383, 385
extended NRBF 92

F

fast Fourier transform 491, 498
feature extraction 2
feature selection
 2, 20, 521, 527, 528, 534, 538
fidelity 404, 406, 421, 426
fitness landscape 96, 108
flexibility test 572
Fourier descriptors 498
frame problem, the 600, 619
fuzzy clustering 2, 22, 23
fuzzy inference system (FIS) 426
fuzzy rule based systems 404, 426

G

Gaussian process (GP) 177, 178, 180, 181,
 182, 187, 188, 189, 190, 191
generalization 521, 530, 531, 533, 535, 538
generative topographic mapping
 176, 192, 193, 194

generative topographic mapping through time (GTM-TT) 176, 177, 178, 179, 180, 181, 182, 183, 185, 187, 188, 189, 190, 191, 192
genetic algorithms 376, 377, 385, 386, 429, 432, 434, 438, 439
genetic programming 379, 383, 385, 386
geographic information resolution (GIRE) 334, 335, 341
geographic information retrieval (GIR) 333, 334, 335, 341
global minimum 567, 573
group technology 20, 23, 27

H

hidden data structures 1, 2
hidden Markov model (HMM) 305, 310, 311, 317, 324, 335, 336, 338, 339, 340, 341, 457, 458, 459, 460, 464, 465, 466, 467, 468, 469, 470, 472, 475, 476, 477, 478, 479, 480
hierarchical clustering 2, 4, 11, 19, 26
hierarchical transfer 247, 248
Hilbert space 224, 225, 241
homonymy 313
Human Genome Project 277
hyperspectral imaging 498
hypersphere 5, 6, 113, 127

I

image segmentation 482, 486, 487, 494, 498
incremental learning 351, 355, 361, 374
inductive bias 244, 245, 246, 257, 260, 264
inductive learning 242, 243, 244, 245, 247, 257, 264
inductive logic programming 374
inductive transfer 245, 246, 248, 257, 259, 260, 261, 262
infinite impulse response (IIR) filters 198, 199, 200, 201, 202, 207, 211, 212, 213, 214, 217, 218, 221
information extraction (IE) 303, 305, 306, 317, 332, 341
instance-based learning 376
isomorphic match 256

iterative linear-quadratic regulator (ILQR) method 205

K

k-means algorithm 4, 5, 28, 29, 34
k-nearest neighbour (kNN) 307, 317, 329, 330, 340, 341

L

latent semantic analysis (LSA) 308, 311, 317
latent variable model 177, 194
learning algorithm 432, 436, 439
limit plane 127
locally connected recurrent neural networks 196, 201, 208, 222
locally recurrent neural networks 195, 200, 201, 214, 215, 217, 218, 219, 222
local minima 567, 573
local search 103, 104, 105, 106, 107, 108
look up table (LUT) 493, 498

M

machine learning (ML) 1, 3, 303, 304, 305, 306, 307, 308, 309, 310, 312, 313, 314, 316, 317, 324, 375, 376, 377, 378, 380, 381, 382, 383, 385, 386
machine vision 486, 488, 492, 494, 495, 496, 497, 498
macro-averaging 306
marginalization 164, 173
Markov chain model (MCM) 332, 341
Markov decision process (MDP) 249, 251, 256, 258
maximisation algorithm 59
maximum likelihood 177, 180, 181
mean squared error (ER) 396
mega-text 325, 326, 340, 347
mega-text language processing 325, 340, 347
meta-predicates 601, 607, 619
micro-averaging 306
milk yield 440, 442, 443, 454, 455, 456
missing at random (MAR) 151, 173, 175
missing completely at random (MCAR) 151, 173, 175

missing data 147, 148, 149, 150, 151, 152, 153, 154, 155, 156, 157, 158, 159, 160, 161, 163, 165, 166, 167, 168, 169, 170, 171, 172, 173
missing data indicator matrix 173
missing data mechanism 151, 173
missing data pattern 173
mixtures-of-experts (ME) 62, 64, 65, 66, 67, 68, 74, 77, 78, 83, 86, 90, 92
model checking 619
model learning 473, 475, 481
model selection 530, 531, 533, 535, 538
motif discovery 300, 301
multilayer perceptron (MLP) 201, 202, 215, 217, 430, 432, 435, 439, 442, 444, 456, 501, 502, 503, 504, 510, 511, 512, 513, 514, 518
multiple imputation 149, 153, 154, 155, 168, 172, 173
multispectral imaging 498
multi-task learning 244, 245, 248, 264
multivariate time series 176, 177, 194
mutual information 520, 526, 527, 533, 538

N

naive Bayes model 244
naïve Bayes (NB) 307, 308, 315, 317, 333, 340, 341
name entity recognition (NER) 311, 317, 333, 337, 341
natural language processing 302, 318, 320, 321, 323, 325, 326, 327, 328, 333, 338, 341, 342, 343, 344, 345, 346, 347
Near InfraRed (NIR) 492, 493, 498
negative characterisation 127
negative transfer 242, 244, 254, 255, 256, 257, 259, 264
nested learning 303, 309, 312, 323
neural gas 6, 13, 14, 23, 27
neural logic network 382, 383, 386
news monitoring 325, 331, 333, 335, 336, 340, 347
nonlinear dimensionality reduction 194
non-local recurrent 196, 209, 211, 212, 213, 222

nonself 125, 127
normalized radial basis function (NRBF) 60, 62, 63, 64, 66, 67, 68, 74, 77, 92
not missing at random (NMAR) 151, 173, 175

O

objective function 95, 96, 97, 101, 104, 108
open loop control system 429, 439
opinion mining 325, 327, 328, 329, 330, 331, 340, 343, 347
orthogonal projection 30
outlier 439
overfitting 430, 436, 439, 521, 528, 530, 531, 533, 535, 538

P

pareto differential evolutionary algorithm 598
parsing 303, 304, 311, 312, 315, 323, 324
particle swarm optimization (PSO) 196, 212
partitional clustering 2, 4, 26
part-of-speech tagging 302, 303, 304, 316, 323
pattern classification 147, 148, 149, 150, 152, 157, 168, 173
pattern evaluation 301
pattern mining 279, 297, 298, 301
phototaxis behavior 575, 576, 579, 582, 585, 586, 598
Poisson's ratio 568, 572
population based algorithm 108
portfolio management 376, 377, 379, 383, 384, 385
positive characterisation 127
positive definite kernel 225, 227, 230, 231, 241
precision agriculture 498
predictive accuracy 173
principal component analysis (PCA) 29, 31, 32, 33, 35, 430, 432
principal components 28, 31, 32, 33, 35, 42, 48, 54, 55, 56, 59
principal cubic complexes 29, 47, 51
principal graph 43, 44, 45, 46, 59

principal manifold 32, 35, 37, 39, 40, 41, 42, 48, 50, 53, 59
 privacy protection 325, 326, 337, 340, 347
 probabilistic calibration for classification 130, 146
 probabilistic calibration for regression 131, 146
 problem distribution 619

Q

Q-learning 265, 266, 268, 269, 272, 273, 276

R

radial basis function network (RBFN) 404, 406, 409, 413, 414, 416, 417, 418, 419, 420, 421, 426, 501, 502, 503, 504, 514, 518
 radial basis function (RBF) 60, 61, 64, 65, 66, 78, 85, 86, 87, 89, 90, 91, 92
 radio frequency localization behavior 598
 real time operation 498
 recurrent neural networks 195, 196, 200, 201, 205, 208, 214, 215, 216, 217, 218, 219, 221, 222, 456
 regularization 528, 538
 reinforcement learning 242, 243, 249, 255, 256, 257, 258, 259, 260, 261, 262, 263, 264, 265, 275, 276, 376, 384
 reliability diagrams 146
 rival penalized competitive learning (RPCL) 60, 66, 67, 74, 77, 81, 90, 92
 rule extraction 404, 405, 406, 413, 422, 426

S

sales forecasting xvii, 387, 389, 396, 397, 399, 400, 401, 403
 SARIMA (seasonal autoregressive integrated moving average model) 387, 388, 396, 397, 399, 403
 search algorithms 301
 search and rescue (SAR) 575, 598
 segmental HMM 460, 464, 465, 469
 self-consistent approximation 59
 self-nonsel discrimination 125, 127

self-organizing feature map (SOFM) 6, 11, 12, 14, 19, 26
 self-organizing maps (SOM) 29, 35, 36, 177, 193, 448, 449, 450, 451, 452, 454, 456
 semantic structures 457, 458, 459, 460, 461, 462, 463, 465, 466, 472, 474, 476, 477, 478, 480
 sequence alignment 282, 301
 sequence motif 301
 shoe upper xxi, 561, 562, 563, 564, 565, 566, 569, 570, 572
 software cost estimation 500, 501, 508, 514, 516, 518
 software inspection 519, 534, 538
 source task 243, 244, 246, 250, 251, 252, 253, 254, 255, 256, 258, 259, 264
 spectral decomposition 33
 sports video analysis 458, 479, 480
 steady-state regime 439
 stem recognition 498
 stochastic algorithm 103, 104, 108
 subspace based functions 60, 62, 78, 79, 86, 92
 support vector clustering (SVC) 5, 6
 support vector machine (SVM) 5, 254, 305, 307, 308, 311, 312, 315, 317, 318, 329, 330, 331, 335, 336, 338, 339, 340, 341, 343, 404, 405, 406, 407, 408, 413, 414, 415, 416, 417, 418, 419, 420, 421, 426, 501, 502, 507, 508, 510, 511, 512, 513, 514, 518
 synaptic weight 207, 212

T

target task 242, 243, 244, 245, 246, 247, 248, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 264
 text classification 325, 326, 331, 340, 342, 347
 time series 389, 390, 391, 392, 393, 394, 396, 397, 398, 399, 400, 403
 transfer learning 242, 243, 244, 245, 246, 248, 257, 258, 259, 260, 261, 262, 264

transient regime 439

traveling salesman problem
4, 17, 18, 24, 25, 27

U

utility problem 606, 607, 619

V

variational Bayesian methods 194
very-large-scale integration (VLSI) 7
video mining 457, 458, 459, 460, 461,
462, 463, 464, 477, 478, 479, 480

W

wavelet representation 230, 241
word-sense disambiguation
302, 303, 304, 315, 323

Y

Young's modulus 568, 572