

PATTERN RECOGNITION AND BIG DATA

Amita Pal

Sankar K Pal



World Scientific

PATTERN RECOGNITION AND BIG DATA

This page intentionally left blank

PATTERN RECOGNITION AND BIG DATA

Amita Pal

Indian Statistical Institute, Kolkata, India

Sankar K Pal

Indian Statistical Institute, Kolkata, India



World Scientific

NEW JERSEY • LONDON • SINGAPORE • BEIJING • SHANGHAI • HONG KONG • TAIPEI • CHENNAI • TOKYO

Published by

World Scientific Publishing Co. Pte. Ltd.

5 Toh Tuck Link, Singapore 596224

USA office: 27 Warren Street, Suite 401-402, Hackensack, NJ 07601

UK office: 57 Shelton Street, Covent Garden, London WC2H 9HE

Library of Congress Cataloging-in-Publication Data

Names: Pal, Amita, editor. | Pal, Sankar K., editor.

Title: Pattern recognition and big data / edited by Amita Pal (Indian Statistical Institute, Kolkata, India), Sankar K Pal (Indian Statistical Institute, Kolkata, India).

Description: [Hackensack?] New Jersey : World Scientific, [2016] |

Includes bibliographical references and index.

Identifiers: LCCN 2016025417 | ISBN 9789813144545 (hardback : alk. paper)

Subjects: LCSH: Big data. | Data mining. | Pattern recognition systems.

Classification: LCC QA76.9.B45 P35 2016 | DDC 005.7--dc23

LC record available at <https://lccn.loc.gov/2016025417>

British Library Cataloguing-in-Publication Data

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

Copyright © 2017 by World Scientific Publishing Co. Pte. Ltd.

All rights reserved. This book, or parts thereof, may not be reproduced in any form or by any means, electronic or mechanical, including photocopying, recording or any information storage and retrieval system now known or to be invented, without written permission from the publisher.

For photocopying of material in this volume, please pay a copying fee through the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923, USA. In this case permission to photocopy is not required from the publisher.

Desk Editor: Amanda Yun

Typeset by Stallion Press

Email: enquiries@stallionpress.com

Printed in Singapore

To our parents and our children

This page intentionally left blank

Preface

The ability to identify patterns is an essential component of sensory intelligent machines. Pattern recognition is therefore an indispensable component of the so-called “Intelligent Control Systems” which involve processing and fusion of data from different sensors and transducers. It is also a necessary function providing failure detection, verification, and diagnosis tasks. Pattern recognition and machine learning form a major area of research and development that encompasses the processing of pictorial and other non-numerical information obtained from interaction between science, technology and society. A motivation for this spurt of activity in this field is the need for the people to communicate with computing machines in their natural mode of communication. Another important motivation is that scientists are also concerned with the idea of designing and making intelligent machines that can carry out certain tasks that humans are adept at, with possibly greater speed and accuracy. This is expected to be the thrust and focus of future generation computing systems.

Machine recognition of patterns is essentially a two-fold task, consisting of learning the invariant and common properties of a set of samples characterizing a class, and of deciding that a new sample is a possible member of the class by noting that it has properties common to those of the set of samples. Therefore, the task of pattern recognition by a computer can be described as a transformation from the measurement space M to the feature space F and finally to the decision space D. Different approaches have emerged as a consequence of application of different paradigms to one or both of the two tasks.

Research in Pattern Recognition has been vigorously pursued over the past several decades, and has witnessed continuous cross-fertilization of ideas from disciplines like computer science, neuro-biology, psychology, physics, engineering, mathematics, statistics and cognitive science. Depending on the need and practical demand, various modern methodologies are being developed which often supplement the classical techniques. Such

an evolution ranges from the decision theoretic (both deterministic and probabilistic), syntactic/ structural, connectionist and fuzzy set theoretic approaches to the more recent soft computing (integration/hybridization of fuzzy sets, artificial neural networks, genetic algorithms and rough sets) approach.

Interest in this field is expected to remain intense in the foreseeable future, in view of the fact that it plays a crucial role in man-machine interaction and automation of cognitive tasks. Moreover, it is of great significance in Data Mining, Knowledge Discovery and Big Data Analytics. These are the main reasons why the editors of this volume have decided to come up with this sequel to an earlier volume on Pattern Recognition, edited by them, which appeared in 2001. The latter contained twenty-one articles, covering both review of older methodologies and more recent approaches, ranging from classical to modern, with significant real-life applications. These articles, written by different experts over the world, demonstrated the significance of this evolution, relevance of different approaches with characteristic features and the formulation of various modern theories. The current volume is a collection of twenty-six articles, contributed by a galaxy of international researchers, providing a fresh look at the development of methodologies and applications, highlighting the paradigm shift that has occurred in the ensuing one and a half decades since the first edition appeared.

This volume starts with an introductory article written by the editors themselves, providing an overview of pattern recognition, different tasks involved, some significant well-established classification techniques as well as the subsequent development in methodologies. The chapter also discusses the significance of data mining, which has captured the imagination of PR researchers world-wide in the past couple of decades. It also addresses the challenges posed by Big Data as well as the relevance of PR and machine learning in this context.

The remaining chapters contain new material mostly, exemplifying how research directions have evolved since the earlier volume. Chapters 2-15 deal with development of methodologies dealing with various aspects of PR tasks as well as different approaches, while the remaining eleven chapters discuss interesting applications. In Chapter 2, Stathopoulos and Girolami have discussed the basics of classification using Gaussian processes (GPs) and reviewed recent advances in approximate inference. For GP classification, they have introduced the Expectation Propagation (EP) algorithm for obtaining an approximation to the posterior distribution that has high

accuracy and have applied it on a real-life problem involving a large number of classes, comparing the performance with support vector machines.

Multitask learning is an approach that learns a problem together with other related issues simultaneously using shared representation. It can reduce the problems with labelled data sparse across different learning task and active learning makes informative queries over unlabelled data to reduce the cost of labelling data. Acharya, Mooney and Ghosh have combined these two aspects in Chapter 3 using latent and supervised shared topics and shown that the combination works satisfactorily in problems where labelled information is expensive to obtain but the learning tasks have some common characteristics.

It is well-known that in pattern recognition and computer vision, the performance of a classifier can be degraded at test time by any distributional change or domain shift in data that occurs after learning. Domain adaptation tries to mitigate this degradation. In Chapter 4, Chellappa and Patel provide an overview of recent developments in domain adaptation for computer vision using sparse and low-rank models, with an emphasis on applications to the problems of face and object recognition. The principle of parsimony or the Occams razor plays a fundamental role in the improvement of the generalization performance in pattern classification. In Chapter 5, Basak provides a quantitative way to express this principle in terms of the outcome of a classifier instead of explicitly regularizing the model complexity in terms of model parameters. The method is then used to design two classifiers: a new kernel machine and a modified k-nearest neighbour algorithm along with their performance evaluation.

Classifier performance gets adversely affected if training data have incorrect labels, that is, if there is label noise in training data, which is quite likely in real-life scenarios. Hence it is always desirable to have classifier design strategies that are robust to noise in training data. In Chapter 6, Sastry and Manwani have provided an overview of various techniques proposed for learning under label noise, after discussing different sources of label noise.

Bahrampour, Nasrabadi, and Ray have focussed on the problem of time-series classification and presented an overview of recent developments in the area of feature extraction and information fusion, in Chapter 7. Specifically, they have reviewed a recently proposed feature extraction algorithm, symbolic dynamic filtering (SDF). The chapter also presents recent developments in the area of sparse-representation-based algorithms for multimodal classification with performance evaluation. Many real-life pattern recog-

nition problems are characterized by imprecision and vagueness, and are difficult to solve with methods based on crisp reasoning. In such cases, methods based on fuzzy logic are more successful. In Chapter 8, Pedrycz and Pizzi focus on this approach, elaborating upon the fundamentals of pattern recognition and fuzzy sets, and the methodological issues. Subsequently, they refer to some concepts pertaining to Granular Computing and information granules, and discuss some representative architectures highlighting the role of fuzzy sets.

Feed-forward neural networks have been found to be very successful in learning patterns during training. However, estimating the required size and structure of the network is very crucial. Gherman, Sirlantzis and Deravi have addressed precisely this problem in Chapter 9. They adopt the divide-et-impera paradigm widely used in computer programming, and show that for certain classes of pattern matching problems, it is possible to identify suitable structures that match the task complexity and thus optimize resource usage while maintaining the performance levels.

Observing that pattern recognition activities, like most problem-solving tasks, involve more than one conflicting objectives, such as accuracy, execution time, and generalization ability, Deb discusses a number of classical and evolutionary multi-criterion optimization methods in Chapter 10. These include the elitist non-dominated sorting genetic algorithm or NSGA-II.8, its comparison with other related methods, and a discussion on handling the number of criteria. As discussed earlier in Chapter 1, the notion of a Rough Set is developed based on the assumption that with every object there is some associated information and the objects are characterized by an indiscernibility relation. In Chapter 11, Skowron, Nguyen and Jankowski present the basic theory of Rough Sets and discuss at length its application in pattern recognition, machine learning and data mining. This chapter not only covers the material presented in the chapter pertaining to this topic in the earlier volume, but also includes information on subsequent development in this area, published after 2001. The authors have also discussed some challenging issues for rough sets and proposed Interactive (Rough) Granular Computing I(R)GC as a framework making it possible to search for solutions of problems related to inducing of relevant contexts, process mining and perception based computing (PBC).

In Chapter 12, Jayadeva, Soman and Chandra discuss the Twin Support Vector Machine (TWSVM), an extension of the widely-used Support Vector Machine (SVM), designed to take care of the inability of the latter to generalize well for imbalanced classification problems. TWSVM minimizes

a linear combination of the structural and empirical risk, finds hyperplanes with small Vapnik-Chervonenkis (VC) dimension, and is faster than SVM in solving optimization problems.

Chapter 13 by Thenkanidiyoor, Dileep and Chandra Sekhar deals with the dynamic kernels-based approaches to classification and matching of varying length patterns. Issues in designing the dynamic kernels, different methods for designing the dynamic kernels; especially the intermediate matching kernel-based dynamic kernel approaches are illustrated. The effectiveness of the approach to analysis of varying-length patterns extracted from speech and image data is demonstrated.

Granular computing provides a paradigm in which computation and operations are performed on complex information entities, called information granules, which are obtained through data abstraction. Integration of this principle with the connectionist approach gives rise to granular neural networks, as discussed in Chapter 1. In Chapter 14, Ganivada, Ray and Pal describe how supervised and unsupervised pattern analysis tasks like classification and clustering can be accomplished using granular neural networks based on fuzzy rough sets. They use multi-layer neural networks and self-organizing map respectively for said tasks. They are superior to other related methods in terms of performance and learning time for overlapping classes.

Maji and Paul present in Chapter 15 a hybrid unsupervised learning algorithm, termed as rough fuzzy c -means (RFCM) algorithm, which integrates the merits of rough sets and the fuzzy c -means algorithm. Its effectiveness is demonstrated in bioinformatics applications RFCM performs better than other c -means algorithms, but it takes more time compared to hard c -means. The algorithm formulation is geared towards maximizing the utility of both rough sets and fuzzy sets with respect to knowledge discovery tasks.

Object detection is one of the most important problems in computer vision and it is the base for many others, such as navigation, stereo matching and augmented reality. One of the most popular and powerful choices for performing object detection is using keypoint correspondence approaches. Several keypoint detectors and descriptors have already been proposed but they often extract information from the neighbourhood of each point individually, without considering the structure and relationship between them. In Chapter 16, Hashimoto, Morimitsu, Hirata-Jr. and Cesar-Jr. have demonstrated how structural pattern recognition techniques can be a powerful tool for filling this gap. They have explored the concept of keygraphs

for extracting structural features from regular keypoints. These result in more flexibility to the description process. The effectiveness of the system has been demonstrated through real-time applications on a mobile phone.

Multimodal data mining refers to analyzing more than one form of data for discovering hidden patterns. In Chapter 17, Chaudhury, Dey, Verma and Hassan have examined different aspects of dealing with big multi-modal data and presented an example of cross-modal information integration between text and image. An application of multi-structured data mining system for business analytics has also been described.

In the next chapter, Wiliem and Lovell address certain issues in the determination of the positive Human Epithelial Type 2 (HEp-2) cells in the Anti-Nuclear Antibodies test to identify the existence of Connective Tissue Diseases through direct Immunofluorescence protocol via image-based analysis. Comparative studies reveal that the traditional method is sufficient for scenarios within a single laboratory but the contemporary approach is more suitable to address scenarios when multiple laboratories are sharing their scanned images and therefore it has potential to replace the current traditional approach in the future.

Reddy, Rama Murty, and Yegnanarayana present in Chapter 19 the development of a spoken term detection system (STD) that searches and locates spoken query words in large volumes of continuous speech. The work focuses on posterior representation of speech for developing the STD system with both supervised and unsupervised approaches. Among the different representation techniques considered, the performance of phonetic posteriors is much better than those by Gaussian Mixture Model and Gaussian Bernoulli Restricted Boltzmann Machine posteriors, but its application is limited since it requires labelled data.

Zhang, in the next chapter, proposes a non-invasive method for detecting Diabetes Mellitus (DM) and Nonproliferative Diabetic Retinopathy (NPDR), the initial stage of Diabetic Retinopathy (DR), from tongue images. Using a combination of 34 features based on color, texture and geometry of tongue images, the method can classify Healthy and DM tongues as well as NPDR tongues and DM tongues without NPDR with reasonably high accuracy.

Subudhi, Ghosh and Ghosh describe a multi-layer Markov model based spatio-temporal segmentation scheme for moving object detection in Chapter 21. Spatial segmentation for initial frame is obtained by multi-layer compound Markov random field (MLCMRF) followed by MAP estimation with a combination of simulated annealing (SA) and iterative conditional

mode (ICM) techniques. For subsequent frames, a change information based heuristic initialization is proposed for faster convergence of the MAP estimation. For temporal segmentation, label difference based change detection technique is used. The method compares favourably with existing state-of-the-art techniques. In the next chapter, Bruzzone, Demir and Bovolo review the recent advances in automatic classification of remote sensing (RS) image time series for updating land-cover maps. Direct supervised classification of each image in the time series is costly and time consuming. To overcome these, domain adaptation (DA) methods have recently been introduced in RS literature. The recent methodological developments related to DA are presented here by focusing on semi-supervised and active learning approaches. Promising methods to define low-cost training sets and to effectively classify RS image time series are also discussed.

In Chapter 23, Sunil, Chaudhuri and Sharma discuss the scope of pattern recognition concepts in Electronic nose (E-nose) application. The functionality of electronic nose is mainly to identify certain target gases through use of a sensor array followed by data acquisition system and pattern recognition algorithm. This chapter, using the data obtained from an E-nose, describes the issues such as feature selection from sensor response, choice of classifier, training of classifier and its validation. The main focus of the work is to develop different methods for optimal selection of sensors for a set of target gas which maximize the classification accuracy.

The next two chapters deal with different aspects of detecting patterns in data being generated from online social networks like Twitter. Rudra, Chakraborty, Ganguly and Ghosh address in Chapter 24 the issue of understanding idioms and its users on Twitter. Specifically, the authors develop two (link-based and content-based) algorithms for community detection on the Twitter social network, and show that idiom-oriented users get clustered better in one while topical users in the other. Finally, they build a novel service which shows trending idioms and recommends idiom users to follow. In the following chapter, Palguna, Joshi, Chakravarthy, Kothari and Subramaniam discuss a theoretical formulation for sampling Twitter data, noting that the analysis of large volumes of Tweets for various applications would require techniques that scale well with the number of Tweets. They introduce metrics to quantify the statistical representativeness or goodness of the samples in terms of restoring public sentiments associated with the frequent keywords, and obtain highly representative Tweet samples. These samples can serve to generate summaries for human use.

In the final chapter, Banerjee and Pal present ideas that envision man-machine symbiosis and aim to contribute to the synthesis of an empathetic artificial mind. Based on Minskys theories of cognition, Zadehs Z -number philosophy, and human brain processes of comprehension, they propose a framework for a machine mind and use a new paradigm, Z^* -number, defined here, to encapsulate the objective and subjective elements of real-world information. The primary novelty in the proposed design lies in the consideration of the machine-self and its derivatives as crucial to true intelligence.

We take this opportunity to thank all the contributors for agreeing to write for the volume. We owe a vote of thanks to Mr. Jason Lim and Ms Amanda Yun of World Scientific Publishing Co. for inviting us to come out with the second edition of our 2001 book. The technical/secretarial assistance provided by Ms Pamli Sengupta and Ms Debasmita Das during the preparation of the manuscript is gratefully acknowledged, as is the support provided by Mr. Taranga Mukherjee and Mr. Anish Mukherjee. The volume is completed while S. K. Pal held the J. C. Bose National Fellowship, the Chair Professorship of the Indian National Academy of Engineering (INAE) and, in part, the Raja Ramanna Fellowship awarded by the Department of Atomic Energy of the Government of India.

*Sankar K. Pal
Amita Pal*

Contents

<i>Preface</i>	vii
1. Pattern Recognition: Evolution, Mining and Big Data <i>A. Pal and S. K. Pal</i>	1
2. Pattern Classification with Gaussian Processes <i>V. Stathopoulos and M. Girolami</i>	37
3. Active Multitask Learning using Supervised and Shared Latent Topics <i>A. Acharya, R. J. Mooney and J. Ghosh</i>	75
4. Sparse and Low-Rank Models for Visual Domain Adaptation <i>R. Chellappa and V. M. Patel</i>	113
5. Pattern Classification using the Principle of Parsimony: Two Examples <i>J. Basak</i>	135
6. Robust Learning of Classifiers in the Presence of Label Noise <i>P. S. Sastry and N. Manwani</i>	167
7. Sparse Representation for Time-Series Classification <i>S. Bahrampour, N. M. Nasrabadi and A. Ray</i>	199

8. Fuzzy Sets as a Logic Canvas for Pattern Recognition <i>W. Pedrycz and N. J. Pizzi</i>	217
9. Optimizing Neural Network Structures to Match Pattern Recognition Task Complexity <i>B. G. Gherman, K. Sirlantzis and F. Deravi</i>	255
10. Multi-Criterion Optimization and Decision Making Using Evolutionary Computing <i>K. Deb</i>	293
11. Rough Sets in Pattern Recognition <i>A. Skowron, H. S. Nguyen and A. Jankowski</i>	323
12. The Twin SVM Minimizes the Total Risk <i>Jayadeva, S. Soman and S. Chandra</i>	395
13. Dynamic Kernels based Approaches to Analysis of Varying Length Patterns in Speech and Image Processing Tasks <i>Veena T., Dileep A. D. and C. Chandra Sekhar</i>	407
14. Fuzzy Rough Granular Neural Networks for Pattern Analysis <i>A. Ganivada, S. S. Ray and S. K. Pal</i>	487
15. Fundamentals of Rough-Fuzzy Clustering and Its Application in Bioinformatics <i>P. Maji and S. Paul</i>	513
16. Keygraphs: Structured Features for Object Detection and Applications <i>M. Hashimoto, H. Morimitsu, R. Hirata-Jr. and R. M. Cesar-Jr.</i>	545

17. Mining Multimodal Data <i>S. Chaudhury, L. Dey, I. Verma and E. Hassan</i>	581
18. Solving Classification Problems on Human Epithelial Type 2 Cells for Anti-Nuclear Antibodies Test: Traditional versus Contemporary Approaches <i>A. Wiliem and B. C. Lovell</i>	605
19. Representation Learning for Spoken Term Detection <i>P. R. Reddy, K. S. R. Murty and B. Yegnanarayana</i>	633
20. Tongue Pattern Recognition to Detect Diabetes Mellitus and Non-Proliferative Diabetic Retinopathy <i>B. Zhang</i>	663
21. Moving Object Detection using Multi-layer Markov Random Field Model <i>B. N. Subudhi, S. Ghosh and A. Ghosh</i>	687
22. Recent Advances in Remote Sensing Time Series Image Classification <i>L. Bruzzone, B. Demir and F. Bovolo</i>	713
23. Sensor Selection for E-Nose <i>Sunil T. T., S. Chaudhuri and M. U. Sharma</i>	735
24. Understanding the Usage of Idioms in Twitter Social Network <i>K. Rudra, A. Chakraborty, N. Ganguly and S. Ghosh</i>	767
25. Sampling Theorems for Twitter: Ideas from Large Deviation Theory <i>D. Palguna, V. Joshi, V. Chakravarthy, R. Kothari and L. V. Subramaniam</i>	789

26. A Machine-mind Architecture and Z*-numbers for Real-world Comprehension	805
<i>R. Banerjee and S. K. Pal</i>	
<i>Author Index</i>	843
<i>Subject Index</i>	845
<i>About the Editors</i>	855

Chapter 1

Pattern Recognition: Evolution, Mining and Big Data

Amita Pal¹ and Sankar K. Pal²

¹*Interdisciplinary Statistical Research Unit
Indian Statistical Institute, Kolkata, India*

pamita@isical.ac.in

²*Machine Intelligence Unit
Indian Statistical Institute, Kolkata, India*
sankar@isical.ac.in

This chapter traces the evolution of pattern recognition (PR) over the years, from its humble beginnings as an extension of statistical discriminant analysis, to the multidisciplinary approach that it has become now, on account of the continuous import of ideas from various scientific disciplines. It begins with an introduction to the discipline of PR, explaining the basic underlying concepts, different tasks involved, some conventional classification techniques and the subsequent development of various modern methodologies. The evolution has been nurtured and aided by the likes of statistical decision theory, the theory of formal languages (which led to the syntactic or structural approach), followed by the theories of fuzzy sets, artificial neural networks, genetic algorithms, rough sets, granular computing and support vector machines individually (leading to different modern approaches), and finally, their integration into the theory of soft computing. While tracing the journey of pattern recognition along this complex route, significant aspects are highlighted. The chapter also discusses the significance of data mining, which has drawn the attention of many PR researchers world-wide for the past couple of decades. Finally, the challenging issues of Big Data analysis are addressed along with the relevance of PR and machine learning.

1.1. Introduction

Pattern recognition is an activity that humans normally excel in. They do it almost all the time, and without conscious effort. Information received

via the various sensory organs is processed almost instantaneously by the human brain so that it is possible to identify the source or content of the information, without any perceptible effort. What is even more impressive is the accuracy with which such recognition tasks can be performed even under non-ideal conditions, for instance, when the information that needs to be processed is vague, imprecise or even incomplete. In fact, most of our day-to-day activities are based on our success in performing various pattern recognition tasks. For example, when we read a book, we recognize the letters, words and, ultimately, concepts and notions, from the visual signals received by our brain, which processes them speedily and probably does a neurobiological implementation of template-matching!

The discipline of Pattern Recognition (PR) or Pattern Recognition by machine essentially deals with the problem of developing algorithms and methodologies/devices that can enable the computer-implementation of many of the recognition tasks that humans normally perform. The motivation is to perform these tasks more accurately, or faster, and perhaps, more economically than humans and, in many cases, to release them from drudgery resulting from performing routine recognition tasks repetitively and mechanically. The scope of PR also encompasses tasks humans are not good at, like reading bar codes. The goal of pattern recognition research is to devise ways and means of automating certain decision-making processes that lead to classification and recognition. PR has been a thriving field of research for the past few decades, as is amply borne out by the numerous books ([1]–[17], for example) and journals devoted exclusively to it. In this regard, mention must be made of the seminal article by Kanal [18] which gives a comprehensive review of the advances made in the field till the early nineteen-seventies. A review article by Jain *et al.* [19] provides an engrossing survey of the advances made in statistical pattern recognition till the end of the twentieth century.

Though the subject has attained a very high level of maturity in the past five decades or so, it remains evergreen to the researchers due to continuous cross-fertilization of ideas from disciplines like computer science, physics, neurobiology, psychology, engineering, statistics, mathematics and cognitive science. Depending on the practical need and demand, various modern methodologies have come into being, which often supplement the classical techniques. The present article gives a bird's-eye view of the different methodologies that have evolved so far including the emergence of data mining. Since any discussion today on pattern recognition remains incomplete without the mention of Big Data, we have included a section

describing the ABCs of Big data, challenging issues, and the relevance of pattern recognition, machine learning and data mining. Before we describe them, we explain briefly the basic concept of PR including supervised and unsupervised classification, and feature selection/extraction. Though these were mentioned in the leading chapter of the first edition of the volume [20], the authors repeat some of them here for the convenience of readers in following the remaining chapters.

1.2. The Pattern Recognition Problem

Let Ω denote the universe of patterns that are of interest, and let \mathbf{X} be a vector of p variables (called features) defined on objects in Ω , which together provide some sort of numerical description for them. Let $\mathcal{X} \subset \mathbb{R}^p$ be the feature space, or the domain of variation of \mathbf{X} corresponding to all patterns in Ω , which contains K categories of objects, where K may or may not be known *a priori*. Let $\Omega_1, \Omega_2, \dots, \Omega_K$ denote the corresponding categories or pattern classes. In this setup, the pattern recognition problem is to determine, for any pattern of unknown categorization (or label) from Ω and having a corresponding feature vector \mathbf{x} , which pattern class it belongs to. Essentially, the general approach to solving the problem is to find, in some way, a partition of \mathcal{X} into $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_K$ so that if $\mathbf{x} \in \mathcal{X}_r$ we can infer that the unknown pattern comes from the pattern class Ω_r . Obviously, this is not possible unless some additional information is provided for each of the classes, say, in the form of a set of n patterns, called training samples.

1.2.1. Supervised vs. unsupervised classification

Human pattern recognition capability is mainly learnt from past experiences, though it is certainly not possible to describe the procedure by which the human brain accomplishes this. Thus learning is an indispensable component of pattern recognizers, both human and mechanical. The information contained in the training samples provides the basis for learning in pattern recognition systems. In some cases, learning is accomplished with the help of a teacher, that is, an external agency of some sort that provides the correct labels or classifications of the training samples for building the classifier. The training samples in such cases become representatives of the classes they belong to, and can be processed in a suitable manner so that the class-specific information they carry may be distilled from them. This is referred to as *supervised pattern recognition*. References [5], [7], [12]–

[17], [21] are a few of the many books in which detailed information on this is available.

On the other hand, if no teacher is available for a pattern classification task, that is, the training samples are not labeled, then we have a case of *unsupervised pattern recognition*. In such cases, learning essentially means discovery of the natural groupings inherent in the training set. The generic name for computational techniques applicable to unsupervised classification is clustering, for which there is no dearth of literature ([1], [2], [22]–[25]).

1.2.2. Feature selection and extraction

Most of the approaches designed for solving PR problems presuppose the representation of patterns by a set of measurements, called features. A judicious selection of features for building classifiers is a very crucial aspect of classifier design, and deserves careful consideration. On one hand, there is certainly nothing to lose in using all available measurements in classifier design. On the other hand, too many features make the classifier increasingly complex (sometimes confusing too), in fact, unnecessarily so, in case some of the measurements are redundant. It is encouraging to see that this aspect of classifier design has indeed been given the importance it deserves, judging from the work reported. References may be found, for example, in [3, 8, 9, 26, 27]. Two broad approaches have been used traditionally. The first is called feature selection, and is essentially the selection of the subset of measurements that optimizes some criterion of separability of classes, since, intuitively, the best set of features should discriminate most efficiently among the classes, that is, enhance the separability among them, while increasing homogeneity within classes at the same time. The other approach, called feature extraction, aims to reduce the number of measurements available in a different way by looking for a transformation of the original vector of measurements that optimizes some appropriately defined criterion of separability among classes, possibly leading to fewer features at the same time.

The following sections give a bird's-eye view of the major developments that have taken place as the discipline of pattern recognition evolved gradually to reach the position it occupies at present. On going through them, the reader is sure to get the impression that PR is thriving and growing at a phenomenal pace, not in the least shy in borrowing methodologies from myriad scientific disciplines in order to provide progressively better techniques over the years.

1.3. Supervised Methods

1.3.1. The statistical approach

When pattern recognition was just beginning to develop as a distinct entity, its practitioners were quick to realize that statistics and probability theory were ideal tools for the task that they had in mind. Statistics could help them to model the inherent variability of patterns in the pattern space via multivariate probability distributions. The classical method of linear discrimination, first proposed by Fisher [28] and later extended by Rao [29] suggested the use of linear combinations of the features, whose coefficients were chosen so as to maximize the ratio of the between-group variance to the within-group variance (the so-called Fisher separability criterion). Such functions, called linear discriminants, could also be constructed using other approaches, like minimum least-squares, linear programming, and so on. A Bayesian decision-theoretic approach for building classification rules was seen to provide an elegant solution to the PR problem in the form of the famous Bayes classifier [3, 5, 16, 30]. It was also possible to build Bayes and other classifiers using nonparametric estimates of the probability density functions involved, in case one did not wish to assume unnecessarily restrictive probabilistic models for the classes. All these seemed to work pretty well in the case of supervised classification, and still do for a multitude of problems. For implementing unsupervised classification, too, statistics has been able to provide tools like clustering and estimation of parameters of mixture densities that are used to model the data in such cases.

For a variety of pattern recognition problems, statistical solutions work reasonably well, both for supervised and unsupervised types. However, there are many other scenarios where they fall short of expectations. Over the years, a multitude of alternative techniques have appeared to take care of such problems. The more noteworthy of these are discussed in some of the following sections. Before moving on to them, let us take a quick look at the major contributions of statistics to PR, that have withstood the test of time. A comprehensive survey of the advances made in statistical pattern recognition till the end of the twentieth century is provided by Jain *et al.* [19].

1.3.1.1. The Bayes decision-theoretic approach

This is essentially a parametric classification procedure, which means that the class-conditional probability densities of the feature vector are assumed

to be of a specific type, say, $f(\mathbf{X}|\Omega_j)$ for the j -th class. Moreover, it is Bayesian, which means that an *a priori* probability distribution is assumed for the parameter of interest which, in this case, is the class label. This associates an *a priori* probability π_j with the j -th class. Reverting to the notation of Section 1.2, we can represent the Bayes classifier for the PR problem mentioned therein, as follows:

Assign an unknown pattern with feature vector \mathbf{x} to the r -th class if

$$p(r|\mathbf{x}) = \max_{j=1,2,\dots,K} p(j|\mathbf{x}),$$

where $p(j|\mathbf{x})$ is the *a posteriori* probability for the class j given the observation \mathbf{x} . It is defined as

$$p(j|\mathbf{x}) = \frac{f_j(\mathbf{x})\pi_j}{\sum_{i=1}^K f_i(\mathbf{x})\pi_i},$$

where $f_i(x) = f(\mathbf{X}|\Omega_i)$ denotes the probability distribution function of the feature vector \mathbf{X} at the point \mathbf{x} in class i , $i = 1, 2, \dots, K$.

It has been shown [5, 7, 11, 16, 30] that this rule has the smallest misclassification probability if one assumes a simple zero-one loss function, and the smallest Bayes risk (that is, the total expected loss assuming both the feature vector and the class label to be random) if we assume a more general loss-function, as compared to rival classifiers for the same problem. Even now the performance of the Bayes classifier for a given PR problem serves as a benchmark against which all other classifiers for the same problem are judged.

1.3.2. Nonparametric approach

In situations where there is not enough prior information available for making distributional assumptions about the feature vector in the different classes, or we do not wish to make too rigorous assumptions about it, then it is possible to take recourse to nonparametric or distribution-free methods in statistics to perform the pattern recognition task. The most obvious way of doing this is to implement parametric classifiers like the Bayes rule, by replacing the probability densities therein by their nonparametric estimates, and then implementing them. A variety of probability density estimators are available in statistical literature, beginning with simple ones like histogram estimators, followed by window estimators, kernel estimators, and so on [3, 5, 11, 21, 31]. The other approach is to use metric-based classification rules like the k -nearest neighbor (k -NN) rules and minimum

distance classifiers [3, 5, 17, 31] that are based on the premise that points in the feature space that are close to each other are very likely to correspond to patterns belonging to the same class. The k -nearest neighbor family of classification rules has certain attractive properties, and certain modifications of it perform almost as well as the Bayes rule.

1.3.2.1. *Discriminant analysis*

The simplest example of this is linear discriminant analysis (LDA), which essentially amounts to approximating boundaries between classes by placing hyperplanes optimally in the p -dimensional feature space among regions corresponding to the different classes, assuming them to be linearly separable. Unknown patterns are classified on the basis of where they lie with respect to the hyperplanes. Since hyperplanes are defined by linear functions of the features (called linear discriminants in this context), essentially this amounts to classifying samples on the basis of the linear discriminant function, disjoint sets of values corresponding to different classes. Classifier design amounts to optimal estimation of the coefficient vector for the linear discriminant on the basis of labeled samples. The crudest way of doing this is to use linear programming techniques to solve a linear optimization problem subject to linear constraints defined by inequalities obtained corresponding to the training samples. More sophisticated methods optimize criteria like mean squared error, among others, [5, 31]. The same idea can easily be generalized to the optimal placement of more complex surfaces corresponding to nonlinear discriminant functions. Discriminant analysis provides a simple yet powerful platform for classification, and the basic principle has been extended and/or modified newer, more powerful methods, for example, as proposed by Bose *et al.* [32]. A lot of recent research in the area has been directed towards making these methods more effective in situations where they fail or are less powerful. One such situation arises when the number of random variables is much larger than the number of observations available on them, as is very often the case with many modern-day real-life problems. In such situations, sample estimates of the class dispersion matrices may be unstable and even singular. Numerous methodologies based on discriminant analysis which are tailor-made for dealing with such situations have been published [33]–[41].

1.3.2.2. Support Vector Machines (SVMs)

These are essentially generalizations of linear decision boundaries for classification. For the case when two classes are linearly separable, optimal separating can be obtained through the application of various principles [5]. For the case of non-separable classes, that is, when the classes overlap, these techniques are generalized to what is known as the support vector machine (SVM), which produces nonlinear boundaries by constructing a linear boundary in a large, transformed version of the feature space.

SVMs have evolved from the so-called maximum margin classifiers for two-class problems, which are separating hyperplanes that have the largest margin, that is, distance from all the training samples. The optimal separating hyperplane obtained thus is determined solely by training samples that are closest, that is least distant from the hyperplane. Under a specific formulation, the maximum margin classifier is obtained by solving a constrained convex optimization problem [5, 31], [42]–[46] using standard Lagrangian methods.

This procedure can be made more flexible by enlarging the feature space using basis functions (φ) such as polynomials or splines. Generally linear boundaries in the enlarged space achieve better training-class separation, and translate to nonlinear boundaries in the original feature space. Once the basis functions are selected, the procedure is the same as before. That is, the maximum margin classifier is determined from the transformed features. The algorithm for finding the maximum margin classifier uses the data only through inner products. Because of this, it can be made non-linear in a very general way, using the so-called kernel trick whereby, instead of mapping the original data via φ and then computing the inner product, inner-product computation in the transformed space can be done in a single step, leaving the mapping completely implicit. In fact, ultimately there is no need to know φ at all; all one needs to know is how to compute the modified inner product, which is referred to as a kernel, $K(x, y)$.

The support vector machine classifier is an extension of this idea, where the dimension of the enlarged space is allowed to get very large, infinite in some cases. It has been found to be highly effective for solving a variety of real-life pattern recognition problems.

1.3.2.3. Classification trees and random forests

Tree-based methods for classification have traditionally been popular in fields like biology and the medical sciences and have the advantage of mak-

ing the classification procedure easy to comprehend. Classification trees are essentially symbolic systems that assign symbolic decisions to examples and are built upon attributes of the patterns that are symbolic in nature or, at the least, discretized if they are not symbolic to begin with. The terminology associated is graph-theoretic, the root being the top node containing all the training samples, which are passed down the tree through successive splits (generally binary), until a terminal node, or leaf, is reached. A split is made at each node on the basis of an optimal value of exactly one of the feature. The variable to be used for splitting, as well as the split point at each node, is determined by maximizing the decrease in impurity at that node. In the classification context, a natural measure of impurity is the misclassification probability. Leaf nodes are pure nodes, or for all practical purposes, have very small impurity values, and are assigned the same class labels as those of the majority of training samples that reach them.

A classification tree partitions the pattern space (or, equivalently, the feature space) into sub-regions corresponding to its leaves, and each unknown pattern is classified by the label of the leaf it reaches. It is not difficult to visualize that a classification tree provides a structured or hierarchical description of the knowledge base. The commonest type of tree-structured classifier is the binary tree classifier in which each non-terminal node has exactly two children. Detailed discussion on this particular approach to classification can be found, for example, in the books by Breiman *et al.* [47], Devroye *et al.* [4], Hastie *et al.* [31] and Ripley [11].

The most crucial aspect of tree-based classification is the automatic construction of trees from a set of examples, that is, tree induction. As can be expected, a variety of algorithms are available in literature, one of the better-known ones being the ID3 of Quinlan [48], as well as its extension, C4.5 [49]. The main differences in the algorithms available for tree construction lie in the rule they use for splitting nodes and in the pruning strategy they use, pruning being the term used to describe the removal of redundant subtrees. For both activities, optimality criteria like information gain or entropy are used. For example, when splitting nodes, the attribute that is used to perform the splitting is selected to be the one for which the optimality criterion is optimized. Similarly, when pruning, the subtree whose removal results in the optimization of the criterion being used, is selected. Many of the algorithms available in literature for tree induction are discussed in [4, 11, 31, 47].

Like many other PR methods, tree-based classification too has imbibed concepts and notions from fuzzy set theory, leading to fuzzy deci-

sion trees [50, 51] that exploit the flexibility in knowledge representation provided by fuzzy logic, in order to deal with vague or imprecise symbolic information. This has, in turn, led to fuzzy versions of many tree-induction algorithms like the ID3 [52].

Random forests [53] are essentially ensemble methods for performing machine learning tasks like classification and regression. They implement ensemble classification by constructing a large number of decision trees during training and returning the mode of the classes as the predicted class label. This has the effect of reducing the tendency of decision trees to overfit to the training data.

The method combines the principle of *bagging* [54] and the random selection of features, proposed independently by Ho [55] and Amit and Geman [56] for constructing a collection of decision trees with controlled variance.

1.3.3. *The syntactic approach*

A significant shortcoming of the statistical and most other subsequent approaches to pattern recognition is that they are ill-equipped to handle contextual or structural information in patterns, that may be very significant in distinguishing among the various pattern classes of interest. In such cases, the pattern can typically be reconstructed from the knowledge of its parts and their interrelationships, and this knowledge may well help in discriminating among pattern classes if the parts and/or interrelationships (that is, the structure) of patterns in one class differs significantly from one class to another. Statistical theory was just not capable to incorporating such complex information as discriminating features. So practitioners of pattern recognition had to search for other approaches which could enable them to model the representation of complex patterns in terms of a recursive decomposition into progressively simpler subpatterns, the interrelationships in the hierarchy of decompositions being well-defined. This was very similar to the problem of decomposing a sentence in some natural language into phrases of different categories which, in turn, are decomposed into words from the vocabulary of the language, using the grammatical rules valid for it (the language). So researchers turned quite naturally to the theory of formal languages [57], and found that much of it was applicable to problems of this type. Hence they used the qualifier *linguistic* or *syntactic* to describe the PR approach that resulted.

Patterns that they were concerned with, were no longer looked upon as arrays of numbers. Rather, these could be described in terms of very simple sub-elements, called primitives, and certain rules, called syntactical or production rules, governing the relationships among them. The collection of primitives and syntactical rules together formed the pattern grammar that characterized the pattern class and specified as completely as possible the patterns that legitimately constitute the class. In order to use this idea for pattern recognition, there should be one grammar for each pattern class, patterns being represented by strings of primitives. The classification strategy was to infer that a given pattern came from a certain class if it belonged to the language (the set of all possible strings of primitives that can be constructed by the grammar) generated by its grammar. The formal procedure that makes this possible is called syntax analysis or parsing. Of course, there was the very important problem of grammatical inference, namely, constructing the syntax rules for each class on the basis of a set of (supervised) training samples (in the form of strings of primitives) belonging to it. Tailor-made procedures for both parsing and grammatical inference were already a part of formal language theory and could easily be applied. Further, the utility, in the context of PR, of various types of automata that serve as recognizers of specific formal languages, was also self-evident.

The approach worked pretty well for idealized patterns, but was, quite naturally, found wanting when it was applied to real-life problems where patterns tended to be noisy and distorted, and hence more complicated, leading very often to ambiguity, namely, a situation where one string or pattern could be generated by more than one grammar. This led, again quite naturally, to the incorporation of probability theory, and later on, fuzzy-set theoretic concepts to model the randomness and vagueness/ imprecision. The outcome of all this activity has been in the form of various types of stochastic and fuzzy grammars and languages [58]–[63]. A comprehensive survey, of these as well as other significant developments in the area, is available in the excellent book by Fu [6]. The approach has great potential, yet has not witnessed as much research activity, particularly after the death of Fu, whose contribution to syntactic PR is universally acknowledged.

1.3.4. *The fuzzy set theoretic approach*

Zadeh [64] proposed a novel approach to the modeling of vagueness and imprecision by means of fuzzy sets, which are generalizations of conventional

(crisp) sets. A fuzzy (sub) set of a universe Ω is defined to be a collection of ordered pairs

$$A = \{(\mu_A(x), x) \mid x \in \Omega\},$$

where $\mu_A(x)$ ($\in [0, 1]$) represents the degree of belonging of the element x to the set A or the degree of its possession of an imprecise property represented by A . Being a generalization of classical set theory, the theory of fuzzy sets allows for greater flexibility in the representation and processing of imprecise or vague information. Various aspects of fuzzy set theory and its relevance to PR are discussed in [1, 2, 8]. It is well-established by now that fuzzy sets can be applied at the feature level to represent input data as an array of membership values denoting the degrees of possession of certain properties; in representing linguistically phrased input features for their processing; in weakening the strong commitments for extracting ill-defined image regions, properties, primitives, and relations among them; and, at the classification level, for representing class memberships of objects, and for providing an estimate of missing information in terms of membership values. In other words, fuzzy set theory provides a notion of embedding, in the sense that a better solution to a crisp problem can be found by visualizing it initially in a wider space, by subjecting it to different (usually fewer) constraints, thus allowing the algorithm greater freedom to avoid errors that are forced upon it by imposition of hard solutions at intermediate levels.

It is worth noting that fuzzy set theory has led to the development of the concept of *soft computing* (to be discussed later) and computational intelligence as a foundation for the conception and design of a high machine IQ (MIQ) system.

The earliest application [66] of the notion of fuzzy sets to supervised pattern recognition was to visualize pattern classes, particularly overlapping ones, as fuzzy subsets of the pattern space. Classification involved *abstraction*, namely, the estimation of the membership functions characterizing the fuzzy pattern classes from the training samples, and *generalization*, which involves imputation of these estimates in the evaluation of the membership values in each class of an unknown pattern. Representation of pattern classes in terms of linguistic features and fuzzy relations is also possible [71, 72]. Decision-theoretic classifiers based on linguistically phrased features have also been proposed [73], as also classification models that incorporate *a priori* knowledge about the classifier from experts

in a linguistic form (Nath *et al.* [74]). Pal and Mandal [75] proposed a multi-valued approach to supervised classification based on approximate reasoning, that can accept imprecise input in linguistic form and provide output in multiple states.

Fuzzy versions of many crisp notions and procedures have been developed, that is, the k -NN rule, decision trees, phrase structure grammars, tree grammars, and so on, which lead to fuzzy forms of classical supervised classification methods. The corresponding references may be found in, for example, [2, 9]. Fuzzy sets have also been used in knowledge-based (KB) approaches to PR that emerged in the beginning of the eighties. In the KB approach, classes are described by rules and the task of recognition is accomplished through automated reasoning or inference procedure. Fuzzy logic can be implemented for describing rules in natural linguistic terms and in fuzzy inferencing with a degree of certainty.

Significant development of fuzzy set theoretic computing paradigms, coined by Zadeh, that have drawn the attention of researchers in the area of cognitive machines, includes computing with words (CWW) [76], computational theory of perceptions (CTP) [77] and *Z-numbers* [78]. Breaking away from the traditional methods of computation using numbers, CWW stands for computation using words and phrases in natural language statements [76]. The paradigm acknowledges the immense cognitive ability of the human brain to formulate decisions on the basis of perceptions of words framing natural language statements. These perceptions may be ambiguous, incorrect or even biased. CTP, as its name implies, deals with computation with perception rather than measurements. Perception may be characterized by f-granulation, that is, the attributes that it can take are granules, while its boundaries are ill-defined (fuzzy) [79]. The *Z-number* [78] has recently been proposed as a model for the precisiation of the perception(s) embedded in a natural language statement [80]. In other words, it provides a qualitative and quantitative abstraction of the deep-semantics in natural language statements. An algorithm for CWW using the *Z-number* approach is described by Pal *et al.* [83] as well as Banerjee and Pal [82] for context granulation and quantification of subjective information [81]. This is applicable to natural language processing (NLP)-like text summarization, semantic disambiguation and concept graph generation. *Z-numbers* have been extended to Z^* -numbers for machine-subjectivity representation by Banerjee and Pal [84].

1.3.5. The connectionist approach

The normal human brain performs countless complex cognitive (and other) tasks effortlessly, accurately and instantaneously by virtue of its architecture, namely, a massively parallel network of biological neurons (or nerve cells), that are basically extremely simple information processing units. Its information processing capability is so robust that the death or malfunctioning of a few of its component neurons generally does not affect its performance perceptibly. Impressed by the efficiency and fault-tolerance of this type of architecture, researchers have tried, over the years, to model information processing systems on it. The models that have emerged from their endeavors, albeit much simpler versions, have succeeded resoundingly in solving a variety of problems, and are called *Artificial Neural Networks* (ANNs) [85]. Basically, each such network consists of numerous densely interconnected simple processing units (or neurons), that is naturally capable of storing prior knowledge and making it available as and when needed. It resembles the human brain in that it acquires knowledge through a learning process and stores the same in a distributed fashion as synaptic weights in the interconnections of the neurons. Numerous books are available on the theory and applications of ANNs [86]–[90], and a number of journals devoted exclusively to various aspects of this relatively new and revolutionary information processing system are in circulation since the early nineties.

Various categories of artificial neural networks (ANNs) have made their appearance over the years, for example, Hopfield networks, Multi-Layer Perceptrons (MLPs), Kohonen self-organizing feature maps (SOFM), Adaptive Resonance Theory (ART) networks, Radial Basis Function (RBF) networks, among others. Some of the tasks that ANNs can perform include supervised pattern classification, clustering, function approximation and optimization. In the context of pattern recognition, it has been established beyond doubt that neural networks are natural classifiers having resistance to noise, tolerance to distorted images or patterns (ability to generalize), superior ability to recognize partially occluded or degraded images or to discriminate among overlapping pattern classes or classes with highly non-linear boundaries, or potential for parallel processing.

ANNs can be viewed as weighted directed graphs in which neurons for nodes and the inter-neuron connections as directed edges. They can be broadly grouped into two categories on the basis of their architectures – feedforward and feedback or recurrent networks. The former are characterized by graphs with no loops, unlike the latter, which have loops on account

of feedback connections. Their adaptability stems from their capacity for learning from their *environments*. There are three broad paradigms of learning – supervised, unsupervised and reinforcement. Various learning algorithms are available in each category. In supervised learning, adaptation is done on the basis of a direct comparison of the network output with the correct or desired label. In unsupervised learning, the network is designed to detect natural groupings in the training set, and forms categories by optimization of some criterion for the quality of clustering induced by the network. Reinforcement learning is looked upon as a special type of supervised learning that tries to learn the input-output relation by optimizing the so-called reinforcement signal. It makes the network aware of the correctness of the decision, but not what the actual decision is. MLPs that learn by *backpropagation of error* [90] have become extremely popular for solving supervised PR problems. Vigorous research activity has been seen since the eighties in the area of pattern recognition by ANNs, and a good overview can be obtained from [9]–[12], [91]. Although ANN research appeared to be somehow in low gear during the past decade, it has regained its momentum recently because of the possibility of its application in Big data analysis from both hardware and software perspectives.

1.3.6. Use of genetic algorithms

Genetic algorithms (GAs) [92]–[97] are randomized search and optimization techniques, inspired by the principle of survival of the fittest governing evolution and selection in natural populations, and are therefore regulated by the laws of genetics. They are capable of obtaining near-optimal solutions by performing efficient, adaptive and robust search, and have parallelism as a major ingredient. In order to approach an optimal solution to a computational problem, a GA starts from a set of assumed solutions likened to and called chromosomes) and evolves different yet better sets of solutions over a sequence of iterations, called generations. In each generation, the objective function (a measure of fitness for survival) determines the suitability of each solution and, on the basis of its values, some of them (called parent chromosomes) are selected for reproduction. Genetic operators like selection/reproduction, crossover and mutation are applied on these and new chromosomes (offsprings) are generated. The number of copies (offsprings) reproduced by an individual parent is expected to be directly proportional to its fitness value. Chromosomes with higher fitness values thus tend to have greater representation in the next generation. Ultimately, after many

generations, only those chromosomes with very high fitness values (corresponding to the optimal solution) proliferate.

Like many other scientific and technical fields of research and development, most approaches for pattern recognition involve the solution of optimization problems of one type of another, thereby making the output dependent on appropriate selection of some parameters. For example, unsupervised classification by clustering involves the optimization of certain objective functions depending upon some parameter values, and so does the induction of tree-structured classifiers. Therefore, GA becomes appropriate and a natural choice for solving many of these problems robustly, and speedily with no fear of getting trapped at local optima. Based on this realization, many researchers have been concentrating on developing GA-based PR methods in the past decade. The books by Pal and Wang [96] as well as Bandyopadhyay and Pal [98] provide an overview of the application of GAs for solving PR-related problems successfully. Some of the approaches, developed later on, using GAs and its improved versions, for example, multi-objective GAs (MOGAs) for solving PR problems are available in [99].

1.3.7. The rough set theoretic approach

The theory of rough sets, as explained by Pawlak [100, 101], has emerged as a major mathematical tool for measuring uncertainty that arises from granularity in the domain of discourse, that is, from the indiscernibility between objects in a set. It has been proved to be useful in a variety of pattern recognition and data mining problems [102, 103]. The theory offers mathematical tools to discover hidden patterns in data, and therefore its importance, as far as pattern recognition and mining is concerned, can in no way be overlooked. A fundamental principle of a rough-set based learning system is to discover the redundancy and dependency between the given features of a classification problem. It approximates a given concept in terms of lower and upper approximations of that concept [100, 101].

The theory of rough sets begins with the notion of an approximation space, which is a pair $\langle U, R \rangle$, where U , the universe of discourse, is a nonempty set and R an equivalence relation on U , that is, R is reflexive, symmetric, and transitive. The relation R decomposes the set U into disjoint classes in such a way that two elements x and y are in the same class iff $(x, y) \in R$. Let U/R denote the quotient set of U by the relation R , and

$$U/R = \{X_1, X_2, \dots, X_i, \dots, X_p\},$$

where X_i is an equivalence class or information granule of R , $i = 1, 2, \dots, p$. If two elements x and y in U belong to the same equivalence class $X_i \in U/R$, we say that x and y are indistinguishable. The equivalence classes of R and the empty set ϕ are the elementary sets in the approximation space $\langle U, R \rangle$. Given an arbitrary set $X \in 2^U$, in general, it may not be possible to describe X precisely in $\langle U, R \rangle$. One may characterize X by its *lower approximation* and its *upper approximation* defined respectively as follows [101]:

$$\underline{R}X = \bigcup_{X_i \in X} X_i,$$

$$\bar{R}X = \bigcup_{X_i \cap X \neq \phi} X_i.$$

Hence, the lower approximation $\underline{R}X$ is the union of all the elementary sets that are subsets of X , and the upper approximation $\bar{R}X$ is the union of all the elementary sets that have a nonempty intersection with X . The interval $\langle \underline{R}X, \bar{R}X \rangle$ is the representation of an ordinary set X in the approximation space $\langle U, R \rangle$ or, simply, the rough sets of X . The lower (respectively, upper) approximation $\underline{R}X$ (respectively, $\bar{R}X$) is interpreted as the collection of those elements of U that definitely (respectively, possibly) belong to X . Further,

- A set $X \in 2^U$ is said to be definable or exact in $\langle U, R \rangle$ iff $\underline{R}X = \bar{R}X$.
- For any $X, Y \in 2^U$, X is said to be roughly included in Y , denoted by $X \tilde{\subset} Y$, iff $\underline{R}X \subseteq \underline{R}Y$ and $\bar{R}X \subseteq \bar{R}Y$.
-

X and Y are said to be roughly equal, denoted by $X \cong_R Y$, in $\langle U, R \rangle$ iff $\underline{R}X = \underline{R}Y$ and $\bar{R}X = \bar{R}Y$.

There are two numerical characterizations of imprecision of a subset X in the approximation space $\langle U, R \rangle$, namely, *accuracy* and *roughness*. The accuracy of X , represented by $a_R(X)$, is the ratio of the number of objects in its lower approximation to that in its upper approximation, that is,

$$a_R(X) = \underline{R}X / \bar{R}X.$$

The roughness of X , denoted by $\rho_R(X)$, is defined as

$$\rho_R(X) = 1 - a_R(X) = 1 - \underline{R}X / \bar{R}X.$$

Therefore, the lower the roughness of a subset X , the better is its approximation.

Rough sets provide an effective tool for extracting knowledge from databases. The method works as follows: First, a knowledge base is created by classifying the objects and attributes within the created decision tables. Then, a knowledge discovery process is initiated to remove some of the undesirable attributes. Finally, the data dependency is analyzed in the reduced database to find the minimal subset of attributes, called *reduct*. A rough set based learning algorithm can be used to obtain a set of rules from a decision table in IF-THEN form [101].

Applications of rough sets in pattern recognition and data mining [102, 103] mainly proceed along the following two directions, namely, decision rule induction from attribute value table and data filtration by template generation. Most of methods of the former category are based on generation of discernibility matrices and reducts. On the other hand, the latter category mainly involves extraction of elementary blocks from the data based on equivalence relation.

The theory of rough sets has been applied successfully to fuzzy rule extraction, reasoning with uncertainty, fuzzy modeling, feature selection, microarray data analysis, prediction of biological activity of molecules, image processing, and web mining, to mention a few. Relevant references can be found in the book by Maji and Pal [103]. A recent application to spatio-temporal outlier detection is available in [104]. The variable precision rough set model [105], multi-granulation rough sets [106], covering-based rough sets [107], tolerant rough sets [108], fuzzy-rough sets [109], generalized rough sets [110], neighborhood rough sets [111], and probabilistic rough sets [112] are the extensions providing new developments of the original (Pawlak's) rough set based knowledge representation. More references to these topics and other applications are available in the books by Pal and Mitra [113], Pal and Shiu [114] as well as Maji and Pal [103].

Two of the important characteristics of rough sets that drew the attention of researchers in pattern recognition and decision science are the capability of uncertainty handling and granular computing. While uncertainty arising from the indiscernibility in the universe of discourse can be handled by the concept of lower and upper approximations of rough sets, granular computing (GrC) is a problem-solving paradigm dealing with the basic elements, called granules, as resulted by equivalence relations. In other words, a granule may be defined as the clump of indistinguishable elements that are drawn together, for example, by indiscernibility, similarity

and functionality. Some of the salient features of granular computing, as relevant for pattern recognition, are explained in the next section.

1.3.8. *Granular computing*

Granulation is a computing paradigm, among others such as self-production, self-organization, functioning of brain, Darwinian evolution, group behavior, cell membranes, and morphogenesis, which are abstracted from natural phenomena. This is inherent in human thinking and reasoning process. This is an important step in the human cognition process.

Granulation can be viewed as the process of construction, representation and interpretation of granules. It involves the process of forming larger objects into smaller and smaller into larger based on the problem in hand. According to Zadeh [79], granulation involves a decomposition of whole into parts. Conversely, organization involves an integration of parts into whole.

Granular computing (GrC) provides an information processing framework where computation and operations are performed on information granules. Information granules are formulated by abstracting the common properties of patterns in data such as similarity, proximity and equality. They represent the structure of patterns evolved by performing operations on the individual patterns. One of the realizations behind GrC is that - precision is sometimes expensive and not much meaningful in modeling and controlling complex systems [79]. When a problem involves incomplete, uncertain and vague information, it may sometimes become difficult to differentiate the individual elements, and one may find it convenient to consider granules to represent a structure of patterns evolved by performing operations on the individual patterns [115, 116]. On the other hand, in some situations although detail information is available, it may be sufficient to use granules in order to have an efficient and practical solution. Accordingly, GrC became an effective framework in designing efficient and intelligent information processing systems for various real life decision-making applications.

Granular computing may be regarded as a unified framework for theories, methodologies, and technique that make use of granules, that is, groups, classes, or clusters of objects in a universe, in the process of problem solving. There are two aspects of granular computing, namely, algorithmic and semantic. In algorithmic aspect, one deals with formation, representation, and interpretation of granules, while the semantic aspect deals with the utilization of granules for problem solving. Since, in granu-

lar computing, computation or operations are performed on granules, that is, on clump of similar or indistinguishable objects or points, rather than on the individual data points, computation time is greatly reduced. From a more practical point of view, the simplicity derived from granular computing is useful for designing scalable pattern recognition and data mining algorithms.

One may note that the construction of granules is a crucial process, as their sizes and shapes are responsible for the success of granular computing based models. Further, the inter- and intra-relationships among granules play an important role. Each of the granules according to its shape and size, and with a certain level of granularity may reflect a specific aspect of the problem. Granules with different granular levels may represent a system differently. Several approaches for granular computing have been suggested in the literature using the paradigms such as fuzzy set theory, rough set theory, power algebras, and interval analysis. For further details on the significance and various applications of GrC, one may refer to [102, 113, 117]–[126].

1.4. Unsupervised Methods

The problem of unsupervised pattern classification may be solved, like problems of various other disciplines, by a method of data analysis known as clustering. The objective of cluster analysis is to partition the given data set into a certain number of natural and homogeneous subsets where the elements of each subset are as similar as possible and dissimilar from those of the other subsets. These subsets are called clusters. The number of such sets may be fixed beforehand or may result as a consequence of some constraints imposed on them. Various algorithms have been developed to date for *clustering* of a data set, that is, to identify the clusters in the data set. Generally they belong to one of two broad categories: partitioning methods and hierarchical methods. A survey of these conventional algorithms is available in many books, for instance, [5, 22–25, 127].

1.4.1. Fuzzy clustering

Vigorous research activity has been seen in the area of clustering based on fuzzy set theory, as is borne out by literature [1, 2, 52, 65]. In all classical clustering algorithms, it is implicitly assumed that disjoint clusters exist in the set of data while in practice, in many cases, the clusters are not com-

pletely disjoint; rather, the separation of clusters is a fuzzy notion. The concept of fuzzy subsets offers special advantage over conventional clustering and allows representation of intractable overlapping configurations of pattern classes. In fuzzy clustering each element is assigned a finite membership to each of the clusters.

The problem of fuzzy clustering was first posed by Ruspini [128] who introduced the notion of a fuzzy partition to represent the clusters in a data set. Zadeh [72] proposed the fuzzy affinity property in order to characterize fuzzy clusters induced by a fuzzy relation R . A new direction in the application of fuzzy set theory to cluster analysis was initiated by Bezdek and Dunn in their work on fuzzy ISODATA [1, 129]. A fuzzy version of the clustering algorithm DYNOC has also been proposed by Pal and Mitra [130]. Backer [65] introduced a clustering model which tries to achieve an optimal decomposition of a group of objects into a collection of induced fuzzy sets by means of a point-to-subset affinity concept, based on their structural properties, among objects in the representation space. Subsequently, fuzzy c -medoids are developed to deal with relational data [103, 131].

A broad overview of the state of the art, as far as fuzzy clustering is concerned, can be had from [2, 52].

1.4.2. *Connectionist methods*

Kohonen Self-organizing Feature Maps (SOFMs) [87, 88] perform unsupervised classification. Learning Vector Quantization (LVQ) is a special case of SOFM learning, both being particular instances of competitive learning, in which the input data is replaced by a much smaller set of prototypes that are good representatives of structure in the data for classifier design.

1.5. The Hybrid Approach and Soft Computing

Integration and further development, in the context of PR, of the aforesaid approaches have been observed for several years. Various hybrid methods have been developed by taking into account the merits of the constituent technologies. A consolidated effort was made to integrate mainly fuzzy logic, artificial neural networks and genetic algorithms, for developing an efficient new paradigm called soft computing, that endows, to an information processing system, the sort of flexibility that is required for handling real-life ambiguous, imprecise situations. The objective is to develop computational paradigms that provide reasonably good solutions at low

cost under non-ideal conditions, and bear close resemblance to human-like decision-making. The salient features of each component have been highlighted in preceding sections. Obviously, a combination of two or more of these methodologies imparts the benefits of each to the hybrid system, leading to more robust solutions. It is no wonder, therefore, that rapid strides are being made in the hybridization of methodologies in the context of pattern recognition. Among the various possibilities, neuro-fuzzy integration [9, 26, 132] is the most visible one. Here the layered network can accept input in fuzzy terms and provides fuzzy output, thereby augmenting its application domain. GAs are used to tune the input membership functions and output decision boundaries, and to determine optimum network parameters. Rough set theory is used for extracting domain knowledge from training samples in the form of crude rules and in encoding initial network parameters for its faster learning [133]–[138]. A survey of the relevance of soft computing to bioinformatics (RNA secondary prediction) has been provided recently by Ray and Pal [139]. A good idea of the current developments in soft computing approaches to PR can be obtained from the recent issues of most PR and machine intelligence journals. Its relation to natural computing is discussed in [123].

In recent years, attempts are being made to integrate fuzzy sets and rough sets judiciously to build a stronger paradigm of uncertainty handling. Here we mention in brief the concept and features of generalized rough sets, developed accordingly.

In Pawlak's rough set theory, both the set X and granules or equivalence relation R are considered to be crisp. However, in real life problems, they could be fuzzy too. Generalized rough sets are defined based on this premise where the expressions for the lower and upper approximations of a set X depend on the type of relation R and whether X is a crisp or a fuzzy set. For example, when R denotes an equivalence relation and X is a crisp set, the pair of sets $\langle RX, \bar{R}X \rangle$ is referred to as the rough set of X (as stated before in Sec. 1.3.7) and $\langle U, R \rangle$ is a crisp equivalence approximation space. When R denotes an equivalence relation and X is a fuzzy set, the pair of fuzzy sets $\langle RX, \bar{R}X \rangle$ is referred to as the rough-fuzzy set of X and $\langle U, R \rangle$ is a crisp equivalence approximation space. When R denotes a fuzzy equivalence relation and X is a crisp set, the pair of fuzzy sets $\langle RX, \bar{R}X \rangle$ is referred to as the fuzzy rough set of X and $\langle U, R \rangle$ is a fuzzy equivalence approximation space. When R denotes a fuzzy equivalence relation and X is a fuzzy set, the pair of fuzzy sets $\langle RX, \bar{R}X \rangle$ is referred to as the fuzzy rough-fuzzy set of X , $\langle U, R \rangle$

being a fuzzy equivalence approximation space. Pictorial diagram of lower and upper approximations for fuzzy rough-fuzzy set of X (that is, the last case) is shown in Fig. 1.1.

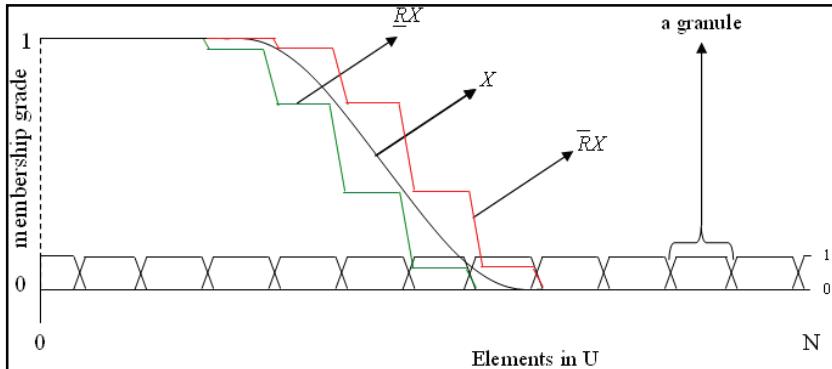


Fig. 1.1. The fuzzy rough-fuzzy set of X .

The significance of generalized rough sets in image analysis is described by Sen and Pal [110], who have defined entropy and image ambiguity measures. Image entropy, formulated on the basis of generalized rough sets, takes care of the uncertainties arising from fuzzy boundaries between regions as well as the rough resemblance between nearby gray levels and nearby pixels in an Image. Therefore, results obtained by rough-fuzzy entropy would be superior to those obtained by either fuzzy entropy or rough entropy. Various approaches of integrating fuzzy sets and rough sets, and their applications in PR problems such as in bioinformatics, medical imaging and neural networks are available in [103, 115, 116, 140]–[142]. Such integration has enhanced the computational intelligence capability of soft computing.

1.6. Data Mining and Knowledge Discovery

The rapid advances being made in computer technology during the past more than a decade have ensured that large sections of the world population have been able to gain easy access to computers on account of falling costs worldwide, and their use is now commonplace in all walks of life. Government agencies, scientific, business and commercial organizations are

routinely using computers not just for computational purposes but also for storage, in massive databases, of the immense volumes of data that they routinely generate, or require from other sources. Large-scale computer networking has ensured that such data has become accessible to more and more people. In other words, we are in the midst of an information explosion, and there is urgent need for methodologies that will help us bring some semblance of order into the phenomenal volumes of data that can readily be accessed by us with a few clicks of the keys of our computer keyboard. Traditional statistical data summarization and database management techniques are just not adequate for handling data on this scale, and for extracting intelligently, information or, rather, knowledge that may be useful for exploring the domain in question or the phenomena responsible for the data, and providing support to decision-making processes. This quest had thrown up some new phrases, for example, data mining [143] and knowledge discovery in databases (KDD), which are perhaps self-explanatory, but will be briefly discussed in the next few paragraphs. Their relationship with the discipline of pattern recognition will also be examined.

The massive databases that we are talking about are generally characterized by the presence of not just numeric, but also textual, symbolic, pictorial and aural data. They may contain redundancy, errors, imprecision, and so on. KDD is aimed at discovering natural structures within such massive and often heterogeneous data. Therefore PR plays a significant role in KDD process. However, KDD is being visualized as not just being capable of knowledge discovery using generalizations and magnifications of existing arid new pattern recognition algorithms, but also the adaptation of these algorithms to enable them to process such data, the storage and accessing of the data, its preprocessing and cleaning, interpretation, visualization and application of the results, and the modeling and support of the overall human-machine interaction. What really makes KDD feasible today is the rapidly falling cost of computation, and the simultaneous increase in computational power, which together make possible the routine implementation of sophisticated, robust and efficient methodologies hitherto thought to be too computation-intensive to be useful. Data mining is that part of knowledge discovery which deals with the process of identifying valid, novel, potentially useful, and ultimately understandable patterns in data, and excludes the knowledge interpretation part of KDD.

Therefore, data mining can be viewed as applying PR and machine learning principles in the context of voluminous, possibly heterogeneous data sets. Furthermore, soft computing-based PR methodologies and ma-

chine learning techniques have been found to be successful in data mining for their ability to handle imprecision, vagueness, uncertainty, approximate reasoning and partial truth and lead to tractability, robustness and low-cost solutions. In this context, case-based reasoning [144] has also played a significant role, as is evident from the books by Pal and Shiu [114] as well as Pal, Dillon and Yeung [137]. Mitra *et al.* [27] provide a comprehensive survey of the relevance of soft computing in data mining tasks. Some of the challenges that researchers in this area have dealt with, include those posed by massive data sets and high dimensionality, nonstandard and incomplete data, and overfitting. The focus had also been on aspects like user interaction, use of prior knowledge, assessment of statistical significance, learning from mixed media data, management of changing data and knowledge, integration of tools, ways of making knowledge discovery more understandable to humans by using rules, and visualization.

1.7. Big Data Analysis

Any discussion on PR and data mining today would remain incomplete without the mention of Big Data.

1.7.1. *What is Big Data?*

It is similar to data as we know it, except that it is substantially bigger in scale, diversity and complexity. However, having the data bigger requires new architectures, techniques, algorithms, tools and analytics to manage it and extract hidden knowledge from it. In doing so, one needs to solve new problems that have cropped up, as well as old problems in a better way. Big Data is characterised by the four Vs, namely, volume (terabytes to zettabytes, 1012-1021), variety (structured, semistructured and unstructured; heterogeneous), velocity (high rate of change, dynamic, streaming) and veracity (uncertainty and incompleteness).

Why is there such a growth in Big Data? The main reasons are:

- Increase in storage capacities
- Increase in processing power
- Availability of data

Data storage has grown significantly after 2000 due to digitalization of analog data, that is, consequent to a shift from analog storage to digital storage. Computation (processing) capability/power has increased sharply

mainly because of the use of mobile phones and video game consoles. According to a recent report in Science by Hibert and Lopez [146], there has been a sharp rise from 25% in 2000 to 94% in 2007 in digital storage among overall storage. Handling-capacity of information by global installed computation has increased from $< 0.001 \times 10^{12}$ million instructions/sec in 1986 to 6.379×10^{12} million instructions/sec in 2007.

The different sectors from where the Big Data is available include: government, communication and media, discrete/ process manufacturing, banking, health-care providers, securities and investment services, education, transportation, insurance, resource industries, and construction. Type of data generated and stored varies with sector. Text/numerals are high in most sectors while video and audio components are in some. For example, in sectors like communication and media, and government, the data has high content of video, audio and text/numbers, and medium amount of image, whereas in manufacturing, it is high for text/numbers, medium for video and image, and low for audio. As expected, in health care, image and text/ numerals are high, and video and audio components are low. All these signify the heterogeneity of the data.

Another source of Big Data is social networks and mobiles. While the use of social networking applications through PCs and smart phones is increasing day by day, the growth is more prominent in case of latter. Interestingly, the number of frequent users has been increasing significantly within the set of all users. Furthermore, the data sets grow in size in part because they are increasingly being gathered by ubiquitous information-sensing mobile devices, Aerial sensory technologies (remote sensing), Software logs, Digital cameras, Microphones, RFID (radio-frequency identification) readers, and Wireless sensor networks.

1.7.2. Dealing with Big Data: challenges and issues

So, Big Data refers to a collection of datasets that grow so large in volume (scalability), variety and velocity (dynamic) and becomes complex that it is difficult to capture, store, manage, share, analyze and visualize it with the conventional data analysis tools. It requires exceptional technologies to efficiently process within tolerable elapsed times. In other words, it needs new forms of processing to enable enhanced decision-making and knowledge discovery, and deliver accurate predictions of various kinds in agile platforms. Here new forms mean new approaches, challenges, techniques, architectures to solve new problems. Accordingly, it demands a

revolutionary change both in research methodologies and tools. Existing computational intelligence techniques may need to be completely re-hauled.

The existing data mining and knowledge discovery processes mainly involve issues related to data modalities like ontologies, structured, networks, text, multimedia and signals, and issues related to data operators like collect, prepare, represent, model, reason and visualize. In case of Big Data, the additional issues include usage, quality, context, streaming and scalability. Typical research areas involved are - information retrieval, pattern recognition, data mining, knowledge discovery in data base, machine learning, natural language processing, semantic web etc. And challenges lie with tasks like - Capturing, Pre-processing, Storage, Search, Retrieval, Analysis and Visualization.

1.7.3. Dealing with Big Data: technologies and uncertainty analysis

Suitable technologies that may be used include crowd sourcing, data fusion and integration, machine learning, signal processing, natural language processing, simulation, time series analysis and visualization. The soft computing (SC) paradigm too appears to have a strong promise in developing methodologies for handling Big Data, as it has been successful in several of these tasks for pattern analysis, data mining and knowledge discovery.

One may note that managing uncertainty in decision-making is very crucial for mining any kind of data, no matter small or big. While fuzzy set (FS) is well known for modelling uncertainty arising from vague, ill-defined or overlapping concepts/ regions, rough set (RS) models uncertainty due to granularity (or limited discernibility) in the domain of discourse. Their effectiveness, both individually and in combination, has been established worldwide for mining audio, video, image and text patterns, as present in the Big Data generated by different sectors. FS and RS can be further coupled, if required, with (probabilistic) uncertainty arising from randomness in occurrence of events in order to result in a much stronger framework for handling real life ambiguous applications. In case of Big-data the problem becomes more acute because of the manifold characteristics of some of the Vs, like high varieties, dynamism, streaming, time varying, variability and incompleteness. This possibly demands the judicious integration of the three aforesaid theories for efficient handling.

While analytics over Big Data is playing a leading role, there has been a shortage of deep analytical talent globally. For example, it has been ob-

served (from McKinsey Global Institute analysis) that the demand in USA is 50 to 60 percent greater than its projected supply by 2018, amounting to a lack of about 140,000 to 190,000 persons. According to sources in the Analytics Special Interest Group set up by NASSCOM there will be a shortage of about 200,000 data scientists in India over the next few years.

In summary, Big Data analysis or analytics research over Big Data has high potential to PR practitioners from the R & D perspective in both academia and industry. It is a highly inter-disciplinary research area comprising statistics, mathematics, computer science, information technology and emerging applications in scientific domain and Industry. The objective is, for example, to develop complex procedures running over large-scale, enormous-sized data repositories for extracting useful knowledge hidden therein, discover new insights from Big Data, and deliver accurate predictions of various kinds as required.

1.8. Conclusion

We have discussed about the various approaches to PR that have emerged so far over the past four decades or so. These range from the statistical, the syntactic, the classification-tree-based, the fuzzy-set-theoretic, the connectionist, the evolutionary-computation-based, rough set theoretic, support vector machine to newly-emerged soft computing (hybrid) and granular computing techniques. Their relevance and salient features are highlighted. These are followed by data mining and knowledge discovery in databases, which has drawn significantly the attention of researchers since 2000, have been explained from the view-point of PR. Finally, the characteristics, issues and challenges of Big Data analysis, which is currently being considered to be the hottest topic of research by data scientists, are discussed. The significance of PR, soft computing and uncertainty analysis therein is explained. As it stands, soft computing methodologies, coupled with case-based reasoning and the computational theory of perception (CTP) [77], the notion of granular computing, and rough-fuzzy approach have great promise for efficient mining of Big data and providing solution of real-life recognition problems. These together would also provide a stronger framework for natural computing. We believe the next decade will bear testimony to this.

Acknowledgement

S.K. Pal acknowledges the J.C. Bose Fellowship of the Government of India, and the Chair Professorship of the Indian National Academy of Engineering (INAE). References

References

- [1] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Functions*. Plenum Press, New York (1981).
- [2] J. C. Bezdek and S. K. Pal, eds., *Fuzzy Models for Pattern Recognition: Methods that Search for Structures in Data*, IEEE Press, New York (1992).
- [3] P. A. Devijver and J. Kittler, *Pattern Recognition: A Statistical Approach*. Prentice-Hall, London (1982).
- [4] L. Devroye, L. Gyorfi and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*. Springer, New York (1996).
- [5] R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification*, 2nd edn. John Wiley & Sons, New York (2000).
- [6] K. S. Fu, *Syntactic Pattern Recognition and Applications*. Prentice-Hall, Englewood Cliffs, NJ (1982).
- [7] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. Morgan Kaufmann, New York (1990).
- [8] S. K. Pal and D. Dutta Majumder, *Fuzzy Mathematical Approach to Pattern Recognition*. Wiley (Halsted Press), New York (1986).
- [9] S. K. Pal and S. Mitra, *Neuro-Fuzzy Pattern Recognition: Methods in Soft Computing*. John Wiley, New York (1999).
- [10] Y. H. Pao, *Adaptive Pattern Recognition and Neural Networks*. Addison-Wesley, Reading, MA (1989).
- [11] B. D. Ripley, *Pattern Recognition and Neural Networks*. Cambridge University Press (1996).
- [12] R. Schalkoff, *Pattern Recognition: Statistical, Structural and Neural Approaches*. John Wiley and Sons, New York. (1992).
- [13] G. S. Sebestyen, *Decision-Making Processes in Pattern Recognition*. McMillan, New York (1962).
- [14] J. Sklansky and G. N. Wassel, *Pattern Classifiers and Trainable Machines*. Springer-Verlag, New York (1981).
- [15] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*. Academic Press, San Diego, CA (1999).
- [16] J. T. Tou and R. C. Gonzalez, *Pattern Recognition Principles*. Addison-Wesley, New York (1974).
- [17] T. Y. Young and T. W. Calvert, *Classification Estimation and Pattern Recognition*. Elsevier, New York (1974).
- [18] L. N. Kanal, Patterns in pattern recognition: 1968-1974, *IEEE Trans. Inf. Theory*, **IT-20**, 697–722 (1974).

- [19] A. K. Jain, R. P. W. Duin and J. Mao, Statistical pattern recognition: a review, *IEEE Trans. Pattern Anal. Mach. Intell.*, **22**, 1–37 (2000).
- [20] S. K. Pal and A. Pal, eds., *Pattern Recognition: From Classical to Modern Approaches*. World Scientific, Singapore (2001).
- [21] D. J. Hand, *Discrimination and Classification*. John Wiley, Chichester (1981).
- [22] M. R. Anderberg, *Cluster Analysis for Applications*. Academic Press, New York (1973).
- [23] B. S. Everitt, S. Landau, M. Leese and D. Stahl, *Cluster Analysis*, 5th edn. John Wiley and Sons, Chichester (2011).
- [24] J. A. Hartigan, *Clustering Algorithms*. Wiley, New York (1975).
- [25] A. K. Jain and R. Dubes, *Algorithms for Clustering Data*. Prentice-Hall, Englewood Cliffs, NJ (1988).
- [26] H. Bunke, and A. Kandel, eds., *Neuro-fuzzy Pattern Recognition*. World Scientific, Singapore (2000).
- [27] P. Mitra, C. A. Murthy, and S. K. Pal, Unsupervised feature selection using feature similarity, *IEEE Trans. Pattern Anal. Mach. Intell.*, **24**(3), 301–312 (2002).
- [28] R. A. Fisher, The use of multiple measurements in taxonomic problems, *Ann. Eugenics*, **7**, 179–188 (1936).
- [29] C. R. Rao, The utilization of multiple measurements in problems of biological classification, *J. Roy. Stat. Soc. B*, **10**, 159–203 (1948).
- [30] T. W. Anderson, *An Introduction to Multivariate Statistical Analysis*, 2nd edn. Wiley, New York (1984).
- [31] T. Hastie, R. Tibshirani and J. Friedman, *The Elements of Statistical Learning*, 2nd edn. Springer (2008).
- [32] S. Bose, A. Pal, R. SahaRay and J. Nayak, Generalized quadratic discriminant analysis, *Patt. Recog.*, **48**(8), 2676–2684 (2015).
- [33] L. Clemmensen, D. Witten, T. Hastie, and B. Ersbøll, Sparse discriminant analysis. *Technometrics*, **53**(4), 406–413 (2011).
- [34] J. Fan, Y. Feng and X. Tong, A road to classification in high dimensional space: the regularized optimal affine discriminant. *J. Roy. Stat. Soc. B*, **74**(4), 745–771 (2012).
- [35] J. H. Friedman, Regularized discriminant analysis, *J. Am. Stat. Assoc.*, **84**(405), 165–175 (1989).
- [36] N. Gkalelis, V. Mezaris and I. Kompatsiaris, Mixture subclass discriminant analysis. *IEEE Signal Process. Lett.*, **18**(5), 319–322 (2011).
- [37] N. Gkalelis, V. Mezaris, I. Kompatsiaris and T. Stathaki, Mixture subclass discriminant analysis link to restricted Gaussian model and other generalizations, *IEEE Trans. Neural Netw. Learn. Syst.*, **24**(1), 8–21 (2013).
- [38] T. Hastie, A. Buja and R. Tibshirani, Penalized discriminant analysis. *Ann. Stat.*, **23**(1), 73–102 (1995).
- [39] Q. Mai, H. Zou and M. Yuan, A direct approach to sparse discriminant analysis in ultra-high dimensions. *Biometrika*, **99**(1), 29–42 (2012).
- [40] J. H. Na, M. S. Park, and J. Y. Choi, Linear boundary discriminant analysis, *Pattern Recogn.*, **43**(3), 929–936 (2010).

- [41] H. Pang, T. Tong and H. Zhao, Shrinkage-based diagonal discriminant analysis and its applications in high-dimensional data. *Biometrics*, **65**(4), 1021–1029 (2009).
- [42] S. Abe, *Support Vector Machines for Pattern Classification*. Springer, London (2010).
- [43] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other Kernel Based Methods*. Cambridge University Press, Cambridge (2000).
- [44] A. Statnikov, C. F. Aliferis and D. P. Hardin, *A Gentle Introduction to Support Vector Machines in Biomedicine, Vol. I: Theory and Methods*. World Scientific, Singapore (2011).
- [45] I. Steinwart, and A. Christmann, *Support Vector Machines*. Springer, Berlin (2008).
- [46] L. Wang, ed., *Support Vector Machines: Theory and Applications*. Springer, Berlin (2005).
- [47] L. Breiman, J. H. Friedman, R. A. Olshen and C. J. Stone, *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA (1984).
- [48] J. R. Quinlan, Induction of decision trees. In *Machine Learning*, 81–106, Kluwer Academic, Boston (1986).
- [49] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann (1993).
- [50] R. L. P. Chang and T. Pavlidis, Fuzzy decision tree algorithms, *IEEE Trans. Syst., Man, Cybern.*, **7**, 28–35 (1977).
- [51] C. Z. Janikow, Fuzzy decision trees: issues and methods, *IEEE Trans. Syst., Man, Cybern. B*, **28**, 1–14 (1998).
- [52] J. C. Bezdek, J. Keller, R. Krishnapuram and N. R. Pal, *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. Kluwer Academic, Boston (1999).
- [53] L. Breiman, Random forests, *Machine Learning*, **45**(1), 5–32 (2001). doi:10.1023/A:1010933404324.
- [54] L. Breiman, Bagging predictors, *Machine Learning*, **24**(2), 123–140 (1996). doi:10.1007/BF00058655.
- [55] T. K. Ho, The random subspace method for constructing decision forests, *IEEE Trans. Pattern Anal. Mach. Intell.* **20**(8), 832–844 (1998). doi:10.1109/34.709601.
- [56] Y. Amit, and D. Geman, Shape quantization and recognition with randomized trees, *Neural Computation*, **9**(7), 1545–1588 (1997).
- [57] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation*. Addison-Wesley, Reading, MA (1979).
- [58] P. R. J. Asveld, Fuzzy context-free languages-Part 1: generalized fuzzy context-free grammars, *Theoretical Computer Science*, **347**(1–2), 167–190 (2005). ISSN 0304-3975, <http://dx.doi.org/10.1016/j.tcs.2005.06.012>.
- [59] P. R. J. Asveld, Fuzzy context-free languages-Part 2: Recognition and parsing algorithms, *Theoretical Computer Science*, **347**(1–2), 191–213, (2005). ISSN 0304-3975, <http://dx.doi.org/10.1016/j.tcs.2005.06.013>.
- [60] G. F. DePalma and S. S. Yau, Fractionally fuzzy grammars with application

- to pattern recognition. In eds. L. A. Zadeh, K. S. Fu, and M. Shimura, *Fuzzy Sets and their Applications to Cognitive and Decision Processes*, Academic, New York. (1975).
- [61] K. S. Fu and T. J. Li, On stochastic automata and languages, *Information Sciences*, **1**, 403–419 (1969).
 - [62] K. Peeva, Finite L-fuzzy acceptors, regular L-fuzzy grammars and syntactic pattern recognition, *Int. J. Uncertainty, Fuzziness and Knowledge-Based Syst.*, **12**(1), 89–104 (2004).
 - [63] M. G. Thomason, Finite fuzzy automata, regular fuzzy languages, and pattern recognition, *Pattern Recogn.* **5**(4), 383–390 (1973).
 - [64] L. A. Zadeh Fuzzy sets, *Inform. Control*, **8**, 338–353 (1965).
 - [65] E. Backer, *Cluster Analysis by Optimal Decomposition of Induced Fuzzy Sets*. University Press, Delft, Holland (1978).
 - [66] R. Bellman, R. Kalaba and L. A. Zadeh, Abstraction and pattern classification, *J. Math. Anal. Appl.*, **13**, 1–7 (1966).
 - [67] A. Kandel, *Fuzzy Mathematical Techniques with Applications*. Addison-Wesley, Reading, MA (1986).
 - [68] G. J. Klir and B. Yuan, *Fuzzy Sets and Fuzzy Logic: Theory and Applications*. Prentice-Hall, Englewood Cliffs, NJ (1995).
 - [69] W. Pedrycz, Fuzzy sets in pattern recognition, *Pattern Recogn.* **23**, 121–146 (1990).
 - [70] L. A. Zadeh, Outline of a new approach to the analysis of complex systems and decision processes, *IEEE Trans. Syst., Man, Cybern.* **3**, 28–44 (1973).
 - [71] L. A. Zadeh, The concept of a linguistic variable and its application to approximate reasoning: Parts 1, 2 and 3, *Inform. Sciences*, **8**, **8** and **9**, 199–249, 301–357 and 43–80 (1975).
 - [72] L. A. Zadeh, *Fuzzy sets and their application to pattern classification and cluster analysis*. Memo no. UCBjERL M-607, University of California, Berkeley (1976).
 - [73] S. K. Pal and D. Dutta Majumder, Fuzzy sets and decision making approaches in pattern recognition, *IEEE Trans. Syst., Man, Cybern.* **7**, 625–629 (1977).
 - [74] A. K. Nath, S. W. Liu and T. T. Lee, On some properties of linguistic classifier, *Fuzzy Set Syst.* **17**, 297–311 (1985).
 - [75] S. K. Pal and D. P. Mandal, Linguistic recognition system based on approximate reasoning, *Inform. Sciences*, **61**, 135–162 (1992).
 - [76] L. A. Zadeh, Fuzzy logic = Computing with words, *IEEE Trans. Fuzzy Syst.*, **4**(2), 103–111 (1996).
 - [77] L. A. Zadeh, A new direction in AI: toward a computational theory of perceptions, *AI Magazine*, **22**, 73–84 (2001).
 - [78] L. A. Zadeh, A note on Z-Numbers, *Inform. Sciences*, **181**, 2923–2932 (2011).
 - [79] L. A. Zadeh, Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic, *Fuzzy Sets Syst.* **90**(2), 111–127 (1997).
 - [80] L. A. Zadeh, Precisiated natural language (PNL), *AI Magazine*, **25**(3), 74–91 (1994).

- [81] S. K. Pal, R. Banerjee, S. Dutta, and S. Sen Sarma, An insight into the Z-number approach to CWW, *Fundamenta Informaticae* (Special issue on Cognitive Informatics and Computational Intelligence), **124**(1–2), 197–229 (2013).
- [82] R. Banerjee and S. K. Pal, On Z-numbers and the machine-mind for natural language comprehension. In eds. D. E. Tamir, N. D. Rishe and A. Kandel, *Fifty Years of Fuzzy Logic and its Applications*, series: Studies in Fuzziness and Soft Computing, **326**, 415–457, Springer (2015).
- [83] S. K. Pal and R. Banerjee, Context granularization and subjective-information quantification, *Theor. Comput. Sci.* **448**, 2–14 (2013).
- [84] R. Banerjee and S. K. Pal, Z^* -numbers: Augmented Z-numbers for Machine-subjectivity Representation, *Inform. Sciences*, **323**, 143–178 (2015).
- [85] R. P. Lippmann, An introduction to computing with neural nets, *IEEE ASSP Magazine*, **4**(2), 4–22 (1987).
- [86] S. Grossberg, ed., *Neural Networks and Natural Intelligence*. The MIT Press, Cambridge, MA (1988).
- [87] T. Kohonen, *Self-Organization and Associative Memory*. 3rd edn. Springer, Berlin (1989).
- [88] T. Kohonen, *Self-organizing Maps*. Springer, Berlin (1995).
- [89] D. E. Rumelhart and J. L. McClelland, eds., *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*, **1**, The MIT Press, Cambridge, MA (1986).
- [90] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation. In eds. D. E. Rumelhart, and J. L. McClelland, *Parallel Distributed Processing: Explorations in the Microstructures of Cognition*. The MIT Press, Cambridge, MA (1986).
- [91] S. K. Pal, and P. K. Srimani, eds., Special issue on Neural Networks: theory and applications, *IEEE Computer*, **19**(3) (1996).
- [92] L. Davis, ed., *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York (1991).
- [93] E. S. Gelsema, ed., Special issue on genetic algorithms, *Pattern Recogn. Lett.* **16**(8) (1995).
- [94] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA (1989).
- [95] M. Mitchell, *An Introduction to Genetic Algorithms*, The MIT Press, Cambridge, MA (1996).
- [96] S. K. Pal and P. P. Wang, eds., *Genetic Algorithms for Pattern Recognition*. CRC Press, Boca Raton, FL (1996).
- [97] S. K. Pal, S. Bandyopadhyay and C. A. Murthy, Genetic algorithms for generation of class boundaries, *IEEE Trans. Syst., Man, Cybern. B: Cybern.*, **28**, 816–828 (1998).
- [98] S. Bandyopadhyay, and S.K. Pal, *Classification and Learning Using Genetic Algorithms: Applications in Bioinformatics and Web Intelligence*. Springer, Berlin (2007).

- [99] S. Bandyopadhyay, S. K. Pal, and B. Aruna, Multiobjective GAs, quantitative indices and pattern classification, *IEEE Trans. Syst., Man, Cybern. B: Cybern.* **34**(5), 2088–2099 (2004).
- [100] Z. Pawlak, Rough sets, *Int. J. Comput. Inf. Sci.* **11**, 341–356 (1982).
- [101] Z. Pawlak, *Rough Sets: Theoretical Aspects of Reasoning about Data*. Kluwer Academic, Dordrecht (1991).
- [102] S. K. Pal and P. Mitra, Case generation using rough sets with fuzzy discretization, *IEEE Trans. Knowl. Data Eng.* **16**(3), 292–300 (2004).
- [103] P. Maji, and S. K. Pal, *Rough-Fuzzy Pattern Recognition: Application in Bioinformatics and Medical Imaging*, Series: Bioinformatics, Computational Intelligence and Engineering, Wiley-IEEE Computer Society Press (2012).
- [104] A. Albanese, S. K. Pal and A. Petrosino, Rough set, kernel set and spatio-temporal outlier detection, *IEEE Trans. Knowl. Data Eng.* **26**(1), 194–207 (2014).
- [105] W. Ziarko, Variable precision rough set model, *J. Comput. Syst. Sci.*, **46**(1), 3959 (1993). ISSN 0022-0000, [http://dx.doi.org/10.1016/0022-0000\(93\)90048-2](http://dx.doi.org/10.1016/0022-0000(93)90048-2).
- [106] Y. Qian, J. Liang, Y. Yao, and C. Dang, MGRS: A multi-granulation rough set. *Inform. Sci.* **180**(6), 949–970 (2010).
- [107] W. Zhu, and F. Y. Wang, On three types of covering-based rough sets, *IEEE Trans. Knowl. Data Eng.* **19**(8), 1131–1144 (2007).
- [108] D. Kim, Data classification based on tolerant rough set. *Pattern Recogn.* **34**(8), 1613–1624 (2001).
- [109] R. Jensen and Q. Shen, Fuzzy-rough sets assisted attribute selection, *IEEE Trans. Fuzzy Syst.* **15**(1), 73–89 (2007).
- [110] D. Sen and S. K. Pal, Generalized rough sets, entropy and image ambiguity measures, *IEEE Trans. Syst., Man, Cybern. B*, **39**(1), 117–128 (2009).
- [111] Q. Hu, D. Yu, J. Liu and C. Wu, Neighborhood rough set based heterogeneous feature subset selection, *Inform. Sci.* **178**, 3577–3594 (2008).
- [112] W. Ziarko, Probabilistic approach to rough sets, *Int. J. Approx. Reason.* **49**(2), 272–284, (2008). ISSN: 0888-613X, <http://dx.doi.org/10.1016/j.ijar.2007.06.014>.
- [113] S. K. Pal and P. Mitra, *Pattern Recognition Algorithms for Data Mining*. Chapman & Hall CRC Press, Boca Raton, FL (2004).
- [114] S. K. Pal and S. C. K. Shiu, *Foundations of Soft Case-based Reasoning*. John Wiley, New York (2003).
- [115] A. Ganivada, S. Dutta and S. K. Pal, Fuzzy rough granular neural networks, fuzzy granules and classification, *Theor. Comput. Sci. C*, **412**(42), 5834–5853 (2011).
- [116] A. Ganivada, S. S. Ray, and S. K. Pal, Fuzzy rough sets, and a granular neural network for unsupervised feature selection, *Neural Networks*, **48**, 91–108 (2013).
- [117] A. Skowron, R. W. Swiniarski, and P. Synak, Approximation spaces and information granulation, *LNCS Trans. Rough Sets*, **3**, 175–189 (2005).
- [118] Y. Li, S. C. K. Shiu and S. K. Pal, Combining feature reduction and case

- selection in building CBR classifiers, *IEEE Trans. Knowl. Data Eng.* **18**(3), 415–429 (2006).
- [119] S. K. Pal, Computational Theory of Perception (CTP), rough-fuzzy uncertainty analysis and mining in bioinformatics and web intelligence: A unified framework, *LNCS Trans. Rough Sets*, **5946**, 106–129 (2010).
 - [120] S. K. Pal, Granular mining and rough-fuzzy pattern recognition: a way to natural computation, (Feature Article), *IEEE Intelligent Informatics Bulletin*, **13**(1), 3–13 (2012).
 - [121] S. K. Pal, S. K. Meher and S. Dutta, Class-dependent rough-fuzzy granular space, dispersion index and classification, *Pattern Recogn.*, **45**(7), 2690–2707 (2012).
 - [122] D. Chakraborty, B. Uma Shankar and S. K. Pal, Granulation, rough entropy and spatiotemporal moving object detection, *Applied Soft Computing*, **13**(9), 4001–4009. (2013).
 - [123] S. K. Pal, and S. K. Meher, Natural computing: a problem solving paradigm with granular information processing, *Applied Soft Computing*, **13**(9), 3944–3955 (2013).
 - [124] W. Pedrycz, *Granular Computing: Analysis and Design of Intelligent Systems*. CRC Press, Boca Raton, FL (2013).
 - [125] J. Yao, A. V. Vasilakos and W. Pedrycz, Granular computing: perspectives and challenges, *IEEE Trans. Cybern.*, **43**, 1977–1989 (2013).
 - [126] S. Kundu and S. K. Pal, FGSN: Fuzzy granular social networks - model and applications, *Information Sciences*, **314**, 100–117 (2015). doi:10.1016/j.ins.2015.03.065.
 - [127] P. H. A. Sneath and R. Sokal, *Numerical Taxonomy*. Freeman, San Francisco, CA (1973).
 - [128] E. H. Ruspini, A new approach to clustering, *Inform. Control*, **15**, 22–32 (1969).
 - [129] J. C. Dunn, A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters, *J. Cybern.* **3**, 32–57 (1973).
 - [130] S. K. Pal and S. Mitra, Fuzzy dynamic clustering algorithm, *Pattern Recogn. Lett.* **11**, 525–535 (1990).
 - [131] P. Maji and S. K. Pal, Rough set based generalized fuzzy C -means algorithm and quantitative indices, *IEEE Trans. Syst., Man, Cybern. B*, **37**(6), 1529–1540 (2007).
 - [132] S. K. Pal, A. Ghosh, and M. K. Kundu, eds., *Soft Computing for Image Processing*. Physica-Verlag, Heidelberg (2000).
 - [133] M. Banerjee, S. Mitra and S. K. Pal, Rough-fuzzy MLP: knowledge encoding and classification, *IEEE Trans. Neural Netw.* **9**, 1203–1216 (1998).
 - [134] S. K. Pal, W. Pedrycz, R. Swiniarski, and A. Skowron, eds., Special issue on Rough-neuro Computing, *Neurocomputing*, **36**(124) (2001).
 - [135] S. K. Pal, and A. Skowron, eds., *Rough-Fuzzy Hybridization: A New Trend in Decision Making*. Springer-Verlag, Singapore (1999).
 - [136] S. K. Pal and A. Skowron, eds., Special issue on Rough Sets, Pattern Recognition and Data Mining, *Pattern Recogn. Lett.*, **24**(6) (2003).
 - [137] S. K. Pal, T. S. Dillon, and D. S. Yeung, eds., *Soft Computing in Case Based Reasoning*. Springer, London (2001).

- [138] S. K. Pal, S. Mitra, and P. Mitra, Rough fuzzy MLP : modular evolution, rule generation and evaluation, *IEEE Trans. Knowl. Data Eng.* **15**(1), 14–25 (2003).
- [139] S. S. Ray and S. K. Pal, RNA secondary structure prediction using soft computing, *IEEE/ACM Trans. Comput. Biol. Bioinf.* **10**(1), 2–17 (2013).
- [140] P. Maji, and S. K. Pal, Feature selection using f -information measures in fuzzy approximation spaces, *IEEE Trans. Knowl. Data Eng.* **22**(6), 854–867 (2010).
- [141] S. K. Pal, and J. Peters, *Rough Fuzzy Image Analysis: Foundations and Methodologies*. CRC Press, Boca Raton, FL (2010).
- [142] S. K. Pal, A. Petrosino and L. Maddalena, eds., *Handbook on Soft Computing for Video Surveillance*. CRC Press, Boca Raton, FL (2012).
- [143] J. G. Shanahan, *Soft Computing for Knowledge Discovery: Introducing Cartesian Granule Feature*. Kluwer Academic, Boston, MA (2000).
- [144] J. L. Kolodner, *Case-Based Reasoning*. Morgan Kaufmann, San Mateo, CA (1993).
- [145] S. Mitra, S. K. Pal, and P. Mitra, Data mining in soft computing framework: a survey, *IEEE Trans. Neural Netw.* **13**(1), 3–14 (2002).
- [146] M. Hibert and P. Lopez, The world's technological capacity to store, communicate, and compute, *Inform. Sci.*, 332 (6025), 60-65 (2011).

Chapter 2

Pattern Classification with Gaussian Processes

Vassilios Stathopoulos* and Mark Girolami†

Department of Statistics

University of Warwick, Coventry, UK

**stathv@gmail.com*, †*m.girolami@warwick.ac.uk*

Recent advances in approximate inference for Gaussian processes (GP) have made possible the efficient implementation of GP models for large classification problems involving large number of classes and large number of observations. In many studies the classification performance of GP classifiers has been shown to be superior compared to other algorithms. Also with GP classifiers it is easy to handle different representations of the data which not need to be in a vectorial form and also combine information from different sources. Nevertheless, GP classifiers are not widely applied in the pattern recognition community.

This chapter offers a tutorial of Gaussian processes for classification and covers recent advances in approximate inference that allow for efficient implementation. The theory behind the models and the algorithms is discussed while detailed pseudo-code is provided in order to allow practitioners to implement the algorithms presented in their own applications.

In the end of this chapter an application on a real problem with large number of classes (21) is presented and results are compared with the popular Support Vector Machine.

2.1. Introduction

In pattern classification we usually consider an input as a vector of D features, $\mathbf{x} \in \mathbb{R}^D$, representing a pattern such as an image. The output of the classifier is a variable $y \in \{1, \dots, C\}$ indicating the class of the input, where C is the total number of possible classes. Most probabilistic classifiers can be described by two functions, the *predictor function* $f(\mathbf{x}; \boldsymbol{\theta})$ and the *link function* $g(\mathbf{z})$. The *predictor function* is subject to unknown

parameters $\boldsymbol{\theta}$ and converts the input \mathbf{x} into "scores". More formally, $\mathbf{f} : \mathbb{R}^D \rightarrow \mathbb{R}^C$. The function values, or scores, $\mathbf{f}(\mathbf{x}; \boldsymbol{\theta}) = [f_1, f_2, \dots, f_C]$ have a very simple interpretation, the higher a value is the more likely the input to belong to the corresponding class. The *link function* $g(\mathbf{z})$ converts the C -dimensional vector of arbitrary real values of scores to a C -dimensional vector of probabilities which sum to 1. The probability of the input vector to belong to the j^{th} class can then be written as

$$p(y = j | \mathbf{x}) = g(\mathbf{f}(\mathbf{x}; \boldsymbol{\theta}))_j.$$

For binary classification problems, where there are two classes, y usually takes values 0 or 1 and there is only one score value. This can be thought as having a single class and we are interested in whether the input belongs to that class or not. For example, in logistic regression $y \in \{0, 1\}$, the predictor function is a linear combination of the input and has the form $f(\mathbf{x}; \boldsymbol{\theta}) = \mathbf{x}^T \boldsymbol{\theta} + \theta_0$, while the link function is the logistic function, $g(z) = 1/(1 + e^{-z})$. The probability of the input belonging to the first class conditioned on the input is then

$$p(y = 1 | \mathbf{x}) = 1/(1 + e^{-(\mathbf{x}^T \boldsymbol{\theta}) + \theta_0}).$$

In classification the form of the *link function* is not very important as it only "squashes" the real score values into probabilities and thus it is usually chosen such that it simplifies the mathematics. On the other hand the form of the *predictor function* and determining the unknown parameters is very important. We can see for example that the linear combination of the input in logistic regression is very limiting since it implies that the class of the inputs can be determined by a separating hyperplane in the \mathbb{R}^D space. Due to this limitation, it is very common to replace the original input with a vector of fixed *basis functions*, $\phi(\mathbf{x})$, which are a non-linear transformation of the input. The *predictor function* will remain linear in the space of basis functions but it will be non-linear in the space of the inputs.

Determining the parameters of the *predictor function* requires a set of N labeled *training inputs* where we have both the inputs \mathbf{x}_n and the corresponding outputs, or class labels y_n , for all $n \in \{1, \dots, N\}$. Then we can find the parameter values which maximise the log likelihood of the data

$$\mathcal{L}(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta}) = \sum_{n=1}^N \log p(y_n | \mathbf{x}_n).$$

However, for linear *predictor functions*, such as the one in linear regression, when the inputs (or the fixed basis transformation of the inputs) in the

training set are completely separable the parameter values that maximise the likelihood tend to infinity. To overcome this problem usually a *regularisation* term, such as the squared L2 norm, is added to likelihood to penalise for very large values. So the parameters are determined by

$$\operatorname{argmax}_{\boldsymbol{\theta}} \left[\mathcal{L}(\mathbf{X}, \mathbf{y}; \boldsymbol{\theta}) - \frac{1}{2} C \|\boldsymbol{\theta}\|^2 \right],$$

where C is a constant controlling the effect of regularisation.

The addition of the regularisation term can be seen as assuming a normal prior distribution with 0 mean and variance C^{-1} for the parameters $\boldsymbol{\theta}$. If we make explicit the dependence of the likelihood function on the parameters and we denote with $p(\boldsymbol{\theta})$ their prior distribution we can rewrite the above expression as

$$\operatorname{argmax}_{\boldsymbol{\theta}} \left[\log p(\boldsymbol{\theta}) + \sum_{n=1}^N \log p(y_n | \mathbf{x}_n, \boldsymbol{\theta}) \right]$$

which we can recognise as the maximum of the log posterior of the parameters $\boldsymbol{\theta}$.

2.2. Gaussian Processes

Instead of trying to choose the right *predictor function*, or choose a suitable set of basis functions, we can work directly on the space of functions. That is, we can specify a *prior over functions* which we will update using the training examples to get a posterior distribution of functions. Of course we cannot work in a computer with the infinite space of functions but we can evaluate them point wise at the inputs we wish.

Gaussian Processes (GPs) can help us achieve exactly this. A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution. A Gaussian Process is solely specified by its mean and covariance functions, $m(\mathbf{x})$ and $k(\mathbf{x}, \mathbf{x}')$ respectively. We can then write that a function is distributed according to a GP as

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')).$$

Drawing a random sample from a Gaussian process is simple since from the definition any number of variables is jointly Gaussian. Thus drawing a random set of function values $\mathbf{f} = [f_1, \dots, f_N]$ for inputs $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_N]$ we first need to compute the covariance matrix \mathbf{K} with elements $K_{i,j} = k(\mathbf{x}_i, \mathbf{x}_j)$, the mean values $\mathbf{m} = m(\mathbf{x}_i)$ and then simply draw a random sample from a multivariate normal with mean \mathbf{m} and covariance \mathbf{K} , i.e.

$\mathbf{f} \sim \mathcal{N}(\mathbf{m}, \mathbf{K})$. Calculating the probability that a set of function values \mathbf{f}' is distributed according to a Gaussian Process with known mean and covariance is also simple since we can calculate the probability density function of a multivariate normal distribution as

$$\mathcal{N}(\mathbf{f}' | \mathbf{m}, \mathbf{K}) = (2\pi)^{-N/2} |\mathbf{K}|^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{f}' - \mathbf{m})^T \mathbf{K}^{-1} (\mathbf{f}' - \mathbf{m})\right).$$

Now consider the case where we know the function values \mathbf{f} for a set of training inputs \mathbf{X} and we want to find the distribution of the function values $\mathbf{f}_* = [f_*^1, \dots, f_*^M]$ for a set of new inputs $\mathbf{X}_* = [\mathbf{x}_*^1, \dots, \mathbf{x}_*^M]$ *conditioned* on the known values. The joint distribution of \mathbf{f} and \mathbf{f}_* is

$$\begin{bmatrix} \mathbf{f} \\ \mathbf{f}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{m} \\ \mathbf{m}_* \end{bmatrix}, \begin{bmatrix} \mathbf{K} & \mathbf{K}_* \\ \mathbf{K}_*^T & \mathbf{K}_{**} \end{bmatrix}\right),$$

where \mathbf{m}_* is simply the mean function values evaluated at the new inputs; i.e. $\mathbf{m}_* = [m(\mathbf{x}_*^1), \dots, m(\mathbf{x}_*^M)]$, \mathbf{K}_* is the covariance of the new inputs and the training inputs; i.e. $(\mathbf{K}_*)_{i,j} = k(\mathbf{x}_i, \mathbf{x}_*^j)$, and \mathbf{K}_{**} is the covariance of the new inputs; i.e. $(\mathbf{K}_{**})_{i,j} = k(\mathbf{x}_*^i, \mathbf{x}_*^j)$. *Conditioning* the multivariate normal on the observed \mathbf{f} then gives us

$$\mathbf{f}_* | \mathbf{X}, \mathbf{f} \sim \mathcal{N}\left(\mathbf{m}_* + \mathbf{K}_*^T \mathbf{K}^{-1} (\mathbf{f} - \mathbf{m}), \mathbf{K}_{**} - \mathbf{K}_*^T \mathbf{K}^{-1} \mathbf{K}_*\right).$$

An example can be seen in Figure 2.1. On the left, samples from the Gaussian Process prior are drawn and we can see that the variance reflects that we do not have any information about the function values. On the

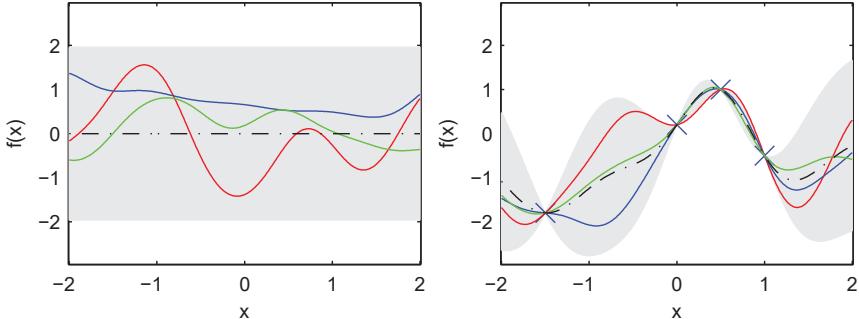


Fig. 2.1. Gaussian Process prior and posterior of functions. Left plot shows the prior distribution and three random samples. Right plot shows posterior distribution and three random samples. Shaded grey region shows the variance of the distribution around the mean (dashed black line). Solid blue, red and green lines are random samples from the GP. Crosses indicate observed values.

right we see how the posterior is updated with the observation of four inputs and the corresponding function values. Notice how the variance around the observed inputs is reduced reflecting our knowledge and that random samples are forced to pass through the observed values. For this example we used a squared exponential covariance function which we will discuss in the next section and a zero mean function.

2.2.1. Covariance functions

As we discussed in the previous section a Gaussian Process is solely specified by its mean and covariance functions. In the classification literature it is common to use a zero mean function which is not necessarily restrictive since the posterior is not restricted to have zero mean. In the remainder of this chapter we will use GP with zero mean functions but in cases where there are specific requirements, such as interpretability or prior assumptions, one can consider incorporating them in a form of a mean function. For more details see [1, Chap. 2.7]. Therefore in GP classification the only requirement is to specify the covariance function. In many pattern classification algorithms there is a notion of similarity between inputs. In the GP classification models this notion is encoded by the covariance function $k(\mathbf{x}, \mathbf{x}')$. But not any function of \mathbf{x} and \mathbf{x}' is a valid covariance.

For practical implementations of a Gaussian Process we use a multivariate normal distribution with a covariance matrix whose elements are values of the covariance function $k(\mathbf{x}, \mathbf{x}')$. In order to be able to draw samples and invert the covariance matrix, for calculating the pdf of the GP, the matrix has to be positive semidefinite (PSD) and thus the covariance function has to be PSD. This is the only requirement that a covariance function must satisfy. In this section we will not discuss all possible covariance functions and their properties. Rather we are going to focus on a few common covariance functions and show how we can interpret their parameters. We will also show how new covariance functions can be created by combining other covariance functions. For more covariance functions and their properties we refer to [1, Chap. 4]

The simplest form of covariance function is the dot product

$$k(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T \mathbf{x}'.$$

Using a GP with this covariance function and a logistic link function to squash the latent function values between 0 and 1 to get the class probability it turns out to be equivalent with the logistic regression model. The classifier is based on a separating hyper plane in the input space and

thus this covariance function is also known as the linear covariance function. Despite its simplicity and restrictions it has proven very useful for high-dimensional problems when for example \mathbf{x} is the vectorised form of a grayscale image with each element corresponding to one pixel.

A more widely used covariance function is the squared exponential function

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp \left(-\frac{\sum_{d=1}^D (x_d - x'_d)^2}{2l^2} \right),$$

where σ and l are the magnitude and length scale parameters. The length scale parameter controls the smoothness of the function and measures, in units, how far on the input space one has to move in order for the function to drastically change value. The magnitude parameter controls the magnitude of this change. Examples of a random sample from a GP with the squared exponential covariance function and different parameter values are shown in Figure 2.2.

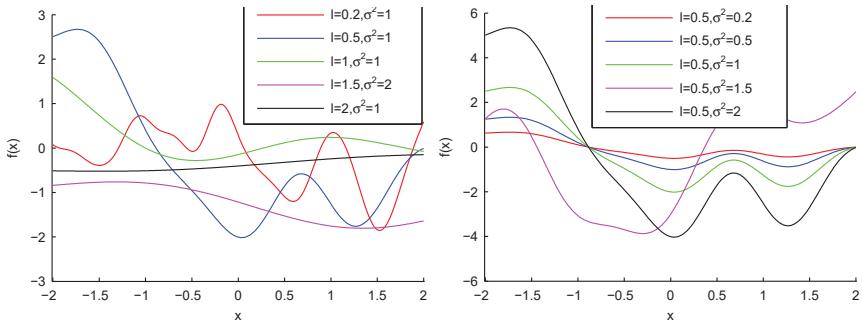


Fig. 2.2. Random samples from a GP with the squared exponential covariance function with different parameters. The random seed is kept fixed for all draws to aid comparison.

The squared exponential function can be generalised to have one length scale parameter for each dimension of the input space, that is

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp \left(-\sum_{d=1}^D \frac{(x_d - x'_d)^2}{2l_d^2} \right).$$

Such a covariance function is also referred to as Automatic Relevance Determination (ARD) and have a very interesting interpretation. Once the optimal parameter values have been found we can inspect their values. As we see in Figure 2.2 the larger the length scale parameter the smoother the function. In the limit of $l_d \rightarrow +\infty$ the function becomes constant and

therefore informs us that there is no information in the particular input dimension x_d for the task at hand.

In many classification problems the input space is not a vector of real values measuring different attributes but it can have a different structure. For example it can be a string in text classification, a tree or even a graph. Instead of converting such representation into a vectorial form we can use covariance functions specifically designed for the input space. One example is the string kernel which can be used when \mathbf{x} and \mathbf{x}' are strings with symbols from a finite alphabet \mathcal{A} ,

$$k(\mathbf{x}, \mathbf{x}') = \sum_{\mathbf{s} \in \mathcal{A}^*} w_{\mathbf{s}} c_{\mathbf{s}}(\mathbf{x}) c_{\mathbf{s}}(\mathbf{x}')$$

where $c_{\mathbf{s}}(\mathbf{x})$ is a function that counts how many times the substring \mathbf{s} appears in \mathbf{x} and $w_{\mathbf{s}}$ is a non-negative weight. Implementation of string covariance functions with linear time complexity in the length of the two strings are possible [2].

Finally, another important benefit of using GP classifiers is also the fact that one can easily combine information from different representations of the inputs or from different sources which not need to have the same structure. That is one can combine string or text information with a vector representation. This can be achieved by combining multiple covariance functions by taking their product or a weighted sum which results in a valid PSD covariance function.

2.3. Pattern Classification with Gaussian Processes

The probabilistic model assumes a *latent* function $\mathbf{f} : \mathbb{R}^D \rightarrow \mathbb{R}^C$ with *latent* values $\mathbf{f}(\mathbf{x}_n) = \mathbf{f}_n := [f_n^1, f_n^2, \dots, f_n^C]^T$ such that when transformed by a sigmoid-like function give the class probabilities $p(y_n | \mathbf{f}_n)$. For a binary classification problem with two classes and with only one latent function value, i.e. $f(\mathbf{x}_n) = f_n$, we can use the probit function.

$$p(y_n | f_n) = \Phi(f_n y_n). \quad (2.1)$$

Whereas in multiple class problems we can use the multinomial probit function [3],

$$p(y_n | \mathbf{f}_n) = \int \mathcal{N}(u_n | 0, 1) \prod_{j=1, j \neq y_n}^C \Phi(u_n + f_n^{y_n} - f_n^j) du_n. \quad (2.2)$$

Notice that there are other alternatives such as the logistic $1/(1 + e^{f_n})$ for the binary case or the softmax function $e^{f_n^{y_n}} / \sum_{j=1}^C e^{f_n^j}$ for multiple classes.

Although the choice of the *link function*^a does not affect the classification results it can make the mathematics easier and thus it indirectly specifies the method that will be used for inference. For example when a logistic or softmax function is used we can perform approximate inference using the Laplace approximation while when the probit or multinomial probit functions are used we can use the Expectation Propagation (EP) algorithm [1, Chap. 3]. In this chapter we will focus on the EP algorithm since it has been shown [4] to provide better approximations to the necessary integrals while recent work has shown that it can efficiently be generalised for the multiple class problem [5].

For the *latent* function values we assume they are *a-priori* independent across classes and that they have zero-mean Gaussian process priors [1, Chap. 3]. Collecting *latent* function values for all inputs and classes in $\mathbf{f} := [f_1^1, \dots, f_N^1, f_1^2, \dots, f_N^2, \dots, f_1^C, \dots, f_N^C]^T$ the GP prior is

$$p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta}) = \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}(\boldsymbol{\theta})), \quad (2.3)$$

where $\mathbf{K}(\boldsymbol{\theta})$ is a $CN \times CN$ block covariance matrix with block matrices $\mathbf{K}^1(\boldsymbol{\theta}), \dots, \mathbf{K}^C(\boldsymbol{\theta})$, each of size $N \times N$, on its diagonal. Elements $K_{i,j}^c$ define the prior covariance between the *latent* function values f_i^c, f_j^c governed by a covariance function $k(\mathbf{x}_i, \mathbf{x}_j | \boldsymbol{\theta})$ with unknown parameters $\boldsymbol{\theta}$.

Optimising the unknown kernel parameters $\boldsymbol{\theta}$ involves computing and maximising the posterior after we marginalise out the unobserved latent function values

$$p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{y}) \propto p(\boldsymbol{\theta}) \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})d\mathbf{f}. \quad (2.4)$$

Making predictions for a new input, y_* , \mathbf{x}_* , involves two steps. First computing the distribution of the *latent* function values for the new input

$$p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\boldsymbol{\theta}}) = \int p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{f}, \mathbf{X}, \hat{\boldsymbol{\theta}})p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \hat{\boldsymbol{\theta}})d\mathbf{f}, \quad (2.5)$$

and then computing the class probabilities using the link function, (or likelihood function in the GP literature),

$$p(y_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\boldsymbol{\theta}}) = \int p(y_*|\mathbf{f}_*)p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\boldsymbol{\theta}})d\mathbf{f}_*. \quad (2.6)$$

^aIn the Gaussian Process literature the *link function* is usually called likelihood function since it is the likelihood of the observed output variables.

2.4. Approximate Inference with Expectation Propagation

Unfortunately exact inference, i.e. computing the integrals, for Equations (2.4–2.6) is not possible and we have to either resort to numerical estimation through Markov Chain Monte Carlo or use approximate methods. Expectation Propagation (EP) [6] is a general algorithm which exploits the factorisation structure of the model for obtaining an approximate posterior distribution. Although there are other alternative approximate inference algorithms for GP classification, such as the Laplace or Variational approximations, the EP algorithm has been shown to provide more accurate approximations to the posterior distribution while recent work [5] has shown that its computational complexity for multi-class problems is similar to other methods, e.g. Laplace.

For Gaussian processes, EP is used to approximate the posterior of the *latent* function values $p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta})$ in Equation (2.5) and the *marginal likelihood* in Equation (2.4). It is also used for computing the integral in (2.6). The posterior distribution can be written as

$$p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \frac{1}{Z} p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta}) \prod_{n=1}^N p(y_n|\mathbf{f}_n), \quad (2.7)$$

where $Z = \int p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta}) \prod_{n=1}^N p(y_n|\mathbf{f}_n) d\mathbf{f}$ is the normalisation term of the posterior, or the *marginal likelihood*.

The EP method approximates the posterior using

$$q_{EP}(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \frac{1}{Z_{EP}} p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta}) \prod_{n=1}^N \tilde{t}_n(\mathbf{f}_n | \tilde{\mathbf{Z}}_n, \tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n), \quad (2.8)$$

where $\tilde{t}_n(\mathbf{f}_n | \tilde{\mathbf{Z}}_n, \tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n) = \tilde{\mathbf{Z}}_n \mathcal{N}(\mathbf{f}_n | \tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n)$ are local *likelihood* approximate terms with parameters $\tilde{\mathbf{Z}}_n, \tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n$. In other words, each *likelihood term* in (2.7) is approximated by a scaled Gaussian function. Notice that the product of the local likelihood approximate terms is a product of *independent* scaled Gaussians and thus it can be written as

$$\prod_{n=1}^N \tilde{t}_n(\mathbf{f}_n | \tilde{\mathbf{Z}}_n, \tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n) = \mathcal{N}(\mathbf{f} | \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}}) \prod_{n=1}^N \tilde{\mathbf{Z}}_n,$$

where $\tilde{\boldsymbol{\Sigma}}$ is a $CN \times CN$ block matrix with blocks $\tilde{\mathbf{Q}}_{i,j}$ $i, j \in \{1, \dots, C\}$ of size $N \times N$ each formed by elements of the local covariance matrices $\tilde{\boldsymbol{\Sigma}}_n$ such that $\tilde{\mathbf{Q}}_{i,j} = \text{diag}([\tilde{\boldsymbol{\Sigma}}_1]_{i,j}, \dots, [\tilde{\boldsymbol{\Sigma}}_N]_{i,j}, \dots, [\tilde{\boldsymbol{\Sigma}}_N]_{i,j})$. Since $p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta})$ is

also a Gaussian then (2.8) is a Normal distribution with covariance $\Sigma = (\mathbf{K}^{-1} + \tilde{\Sigma}^{-1})^{-1}$ and mean $\mu = \Sigma \tilde{\Sigma}^{-1} \tilde{\mu}$.

EP estimates the parameters of the approximate terms by minimising the KL divergence between the posterior and its approximation, $\text{KL}(p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) \| q_{EP}(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}))$. However, direct minimisation is intractable and EP relies on updating the parameters of each approximate term sequentially by matching their moments with the moments of the corresponding exact term. This can be shown to be equivalent to minimising the KL divergence between the posterior and its approximation, see [7, Chap. 10.7] for details.

In more detail, at each iteration the approximation parameters for the n -th term are updated by first computing the *cavity* distribution

$$q_{-n}(\mathbf{f}_n) := q_{EP}(\mathbf{f}_n|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) \tilde{t}_n(\mathbf{f}_n | \tilde{Z}_n, \tilde{\mu}_n, \tilde{\Sigma}_n)^{-1}. \quad (2.9)$$

Then the product of the *cavity* distribution with the exact likelihood term is approximated by another Gaussian, the *tilted* distribution, by matching the zero-th, first and second moments. That is

$$\hat{q}(\mathbf{f}_n) := \hat{Z}_n^{-1} \mathcal{N}(\mathbf{f}_n | \hat{\mu}_n, \hat{\Sigma}_n) \approx q_{-n}(\mathbf{f}_n) p(y_n | \mathbf{f}_n). \quad (2.10)$$

$$\begin{aligned} \hat{Z}_n &= \int q_{-n}(\mathbf{f}_n) p(y_n | \mathbf{f}_n) d\mathbf{f}_n \\ \hat{\mu}_n &= \int \mathbf{f}_n q_{-n}(\mathbf{f}_n) p(y_n | \mathbf{f}_n) d\mathbf{f}_n \\ \hat{\Sigma}_n &= \int \mathbf{f}_n \mathbf{f}_n^T q_{-n}(\mathbf{f}_n) p(y_n | \mathbf{f}_n) d\mathbf{f}_n - \hat{\mu}_n \hat{\mu}_n^T \end{aligned}$$

Finally the approximation parameters are updated such that the product of the cavity distribution and the approximate term \tilde{t}_n have the desired moments. This can be done by equating $\tilde{Z}_n, \tilde{\mu}_n, \tilde{\Sigma}_n$ with the corresponding moments of $\hat{q}(\mathbf{f}_n) q_{-n}(\mathbf{f}_n)^{-1}$ and since both $\hat{q}(\mathbf{f}_n)$ and $q_{-n}(\mathbf{f}_n)$ are Gaussian the moments are in closed form. The algorithm repeats updating the parameters sequentially until convergence.

2.4.1. The binary case

For binary classification with two classes where $y \in \{0, 1\}$ there is only one latent function value per input and thus the vector of latent values is $\mathbf{f} := [f_1, \dots, f_n]^T$. Using the probit likelihood function the posterior is then

$$p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \frac{1}{Z} \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}) \prod_{n=1}^N \Phi(y_n f_n), \quad (2.11)$$

which is approximated using univariate scaled Gaussian functions

$$q_{EP}(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \frac{1}{Z_{EP}} \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}) \prod_{n=1}^N \tilde{Z}_n \mathcal{N}(f_n|\tilde{\mu}_n, \tilde{\sigma}_n^2) = \mathcal{N}(\mathbf{f}|\boldsymbol{\mu}, \boldsymbol{\Sigma}). \quad (2.12)$$

In (2.12) $\boldsymbol{\Sigma} = (\mathbf{K}^{-1} + \tilde{\boldsymbol{\Sigma}}^{-1})^{-1}$ and $\boldsymbol{\mu} = \boldsymbol{\Sigma} \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\boldsymbol{\mu}}$ where $\tilde{\boldsymbol{\Sigma}}$ is a $N \times N$ diagonal matrix with elements $\tilde{\sigma}_n^2$ on the main diagonal and $\tilde{\boldsymbol{\mu}} = [\tilde{\mu}_1, \dots, \tilde{\mu}_N]^T$.

The cavity distribution for the n -th term can be obtained by taking the approximate posterior marginal and dividing it with the corresponding approximate term and since both are Gaussians the result is also a Gaussian of the form

$$q_{-n}(f_n) = \mathcal{N}(f_n|\mu_{-n}, \sigma_{-n}^2)$$

with $\sigma_{-n}^2 = (\sigma_n^{-2} - \tilde{\sigma}_n^{-2})^{-1}$ and $\mu_{-n} = \sigma_{-n}^2(\sigma_n^{-2}\mu_n - \tilde{\sigma}_n^{-2}\tilde{\mu}_n)$ where we have used the fact that the marginal approximate posterior variance for latent variable f_n is $\sigma_n^2 = [\boldsymbol{\Sigma}]_{n,n}$.

The moments of the n -th tilted distribution can be obtained analytically by matching the moments of the cavity distribution multiplied by the corresponding n -th probit term. That is

$$\hat{q}(f_n) := \hat{Z}_n^{-1} \mathcal{N}(\mathbf{f}_n|\hat{\mu}_n, \hat{\sigma}_n^2) \approx \mathcal{N}(f_n|\mu_{-n}, \sigma_{-n}^2) \Phi(f_n y_n). \quad (2.13)$$

$$\begin{aligned} \hat{Z}_n &= \Phi(z_n), \\ \hat{\mu}_n &= \mu_{-n} + \frac{y_n \sigma_{-n}^2 \mathcal{N}(z_n)}{\Phi(z_n) \sqrt{1 + \sigma_{-n}^2}}, \\ \hat{\sigma}_n^2 &= \sigma_{-n}^2 - \frac{\sigma_{-n}^4 \mathcal{N}(z_n)}{(1 + \sigma_{-n}^2 \Phi(z_n))} \left(z_n + \frac{\mathcal{N}(z_n)}{\Phi(z_n)} \right), \\ z_n &= \frac{y_n \mu_{-n}}{\sqrt{1 + \sigma_{-n}^2}}. \end{aligned}$$

The integrals can be solved analytically due to the properties of the univariate normal and probit functions. For a detailed derivation of the moments see [1, Chapter 3.9].

Finally, the approximate parameters for the n -th term are updated such that they have the desired moments by equating them with the moments of $\hat{q}(f_n) q_{-n}(f_n)^{-1}$,

$$\tilde{\sigma}_n^2 = (\hat{\sigma}_n^{-2} - \sigma_{-n}^{-2})^{-1}, \quad (2.14)$$

$$\tilde{\mu}_n = \tilde{\sigma}_n^2 (\hat{\sigma}_n^{-2} \hat{\mu}_n - \sigma_{-n}^{-2} \mu_{-n}), \quad (2.15)$$

$$\tilde{Z}_n = \hat{Z}_n \mathcal{N}(\mu_{-n}|\tilde{\mu}_n, \sigma_{-n}^2 + \tilde{\sigma}_n^2)^{-1}. \quad (2.16)$$

For optimising the covariance hyper-paramarameters θ we need to maximise the marginal posterior in Equation (2.4) which involves the marginal likelihood. The normalisation term of the EP approximation in Equation (2.12) gives us an approximation to the marginal likelihood which we can use for finding the hyper-parameters. The un-normalised EP posterior can be written as

$$\mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}) \mathcal{N}(\mathbf{f}|\tilde{\mu}, \tilde{\Sigma}) \prod_{n=1}^N \tilde{Z}_n.$$

Taking the product of the two Gaussians gives us

$$\mathcal{N}(\mathbf{f}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \mathcal{N}(\tilde{\boldsymbol{\mu}}|\mathbf{0}, \mathbf{K} + \tilde{\boldsymbol{\Sigma}}) \prod_{n=1}^N \tilde{Z}_n$$

thus the EP normalisation term is

$$Z_{EP} = \mathcal{N}(\tilde{\boldsymbol{\mu}}|\mathbf{0}, \mathbf{K} + \tilde{\boldsymbol{\Sigma}}) \prod_{n=1}^N \tilde{Z}_n. \quad (2.17)$$

For making predictions we first need to calculate the predictive distribution, Equation (2.5), of the latent variables for new inputs \mathbf{x}_* . The first term in the integral of (2.5) is the conditional distribution of the GP prior for the new latent variable which from the properties of the multivariate normal distribution we know it is also a Normal of the form $p(f_*|\mathbf{x}_*, \mathbf{f}, \mathbf{X}, \hat{\theta}) = \mathcal{N}(f_*|\mu_*, \sigma_*^2)$ with $\mu_* = \mathbf{k}_*^T \mathbf{K}^{-1} \mathbf{f}$ and $\sigma_*^2 = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^T \mathbf{K}^{-1} \mathbf{k}_*$ where \mathbf{k}_* denotes the vector of covariances between the new input and all inputs in the training set and $k(\mathbf{x}_*, \mathbf{x}_*)$ the variance for the new input. The second term in the integral is the posterior of the latent variables which we approximate using the EP algorithm. Since the approximate posterior and the conditional distribution are both normals then the predictive distribution will also be Normal (see Appendix A.1 for details) with mean and variance

$$E_q[f_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\theta}] = \mathbf{k}_*^T (\mathbf{K} + \tilde{\boldsymbol{\Sigma}})^{-1} \tilde{\boldsymbol{\mu}} \quad (2.18)$$

$$\text{Var}_q[f_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\theta}] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^T (\mathbf{K} + \tilde{\boldsymbol{\Sigma}})^{-1} \mathbf{k}_* \quad (2.19)$$

The predictive distribution for the latent variable is then used in Equation (2.6) for calculating the predictive distribution of the new output y_* . The integral has the same form as the integral used to calculate the zeroth moment (\hat{Z}_n) of the *tilted* distribution above, that is, is the integral of a normal multiplied by a probit. The approximate predictive probability for the new output is then

$$q_{EP}(y_* = 1|\mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\theta}) = \Phi \left(\frac{E_q[f_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}]}{\sqrt{1 + \text{Var}_q[f_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}]}} \right).$$

2.4.1.1. Implementation

For the sake of completeness we describe a practical and efficient implementation of the EP algorithm for the binary problem as presented in [1, Chapter 3.6]. The EP algorithm is given in pseudo-code in Algorithm 2.1.

Algorithm 2.1 EP algorithm for binary classification

```

1: procedure EP_BINARY( $\mathbf{K}, \mathbf{y}$ )
2:    $\tilde{\mathbf{v}} = 0$ ,  $\tilde{\mathbf{t}} = 0$ ,  $\Sigma = \mathbf{K}$ ,  $\mu = 0$ 
3:   repeat
4:     for  $n := 1$  to  $N$  do
5:        $t_{-n} = \sigma_n^{-2} - \tilde{t}_n$                                  $\triangleright$  Compute cavity
6:        $v_{-n} = \sigma_n^{-2} \mu_n - \tilde{v}_n$ 
7:       Compute tilted moments  $\hat{\mu}_n, \hat{\sigma}_n^2$  using Equations (2.14–2.15)
8:        $\Delta\tilde{t}_n = \hat{\sigma}_n^{-2} - t_{-n} - \tilde{t}_n$ 
9:        $\tilde{t}_n = \tilde{t}_n + \Delta\tilde{t}_n$                                  $\triangleright$  Update parameters
10:       $\tilde{v}_n = \hat{\sigma}_n^{-2} \hat{\mu}_n - v_{-n}$ 
11:       $\triangleright$  Update approximate mean and covariance
12:       $\Sigma = \Sigma - ((\Delta\tilde{t}_n)^{-1} + \Sigma_{n,n})^{-1} \tilde{\mathbf{t}}_n \tilde{\mathbf{t}}_n^T$ ,  $\mu = \Sigma \tilde{\mathbf{v}}$ 
13:   end for
14:    $\triangleright$  Re-compute approximate mean and covariance for stability
15:    $\mathbf{L} = \text{Chol}(\mathbf{I} + \tilde{\mathbf{T}}^{1/2} \mathbf{K} \tilde{\mathbf{T}}^{1/2})$ ,  $\mathbf{V} = \mathbf{L}^T \backslash \tilde{\mathbf{T}}^{1/2} \mathbf{K}$ 
16:    $\Sigma = \mathbf{K} - \mathbf{V}^T \mathbf{V}$ ,  $\mu = \Sigma \tilde{\mathbf{v}}$ 
17:   until convergence
18:    $\log |\mathbf{I} + \tilde{\mathbf{T}} \mathbf{K}| = \sum_{n=1}^N \log(\mathbf{L})_{n,n}$ 
19:    $\log Z_{EP} = \frac{1}{2} \tilde{\mathbf{v}}^T \mu - \log |\mathbf{I} + \tilde{\mathbf{T}} \mathbf{K}| + \frac{1}{2} \sum_{n=1}^N (v_{-n}^2 t_{-n}^{-1} + \log t_{-n}^{-1})$ 
20:    $- \frac{1}{2} \sum_{n=1}^N (\mu_n^2 \sigma_n^{-2} + \log \sigma_n^2) + \sum_{n=1}^N \log \hat{Z}_n$ 
21:   return  $\tilde{\mathbf{t}}, \tilde{\mathbf{v}}$  and  $\log Z_{EP}$ 

```

Firstly we note that is more convenient to work with the *natural* approximate parameters and thus the following transformations are used

$$\begin{aligned} \tilde{t}_n &= \tilde{\sigma}_n^{-2}, & \tilde{\mathbf{T}} &= \text{diag}(\tilde{\mathbf{t}}), & \tilde{\mathbf{v}} &= \tilde{\mathbf{T}} \tilde{\mu}, \\ t_{-n} &= \sigma_{-n}^{-2}, & \mathbf{T} &= \text{diag}(t_{-n}), & v_{-n} &= t_{-n} \mu_{-n}. \end{aligned}$$

The covariance matrix of the approximate posterior in (2.8) can now be written as

$$\Sigma = (\mathbf{K}^{-1} + \tilde{\mathbf{T}})^{-1} = \mathbf{K} - \mathbf{K}(\mathbf{K} + \tilde{\mathbf{T}}^{-1})^{-1} \mathbf{K} = \mathbf{K} - \mathbf{K} \tilde{\mathbf{T}}^{1/2} \mathbf{B}^{-1} \tilde{\mathbf{T}}^{1/2} \mathbf{K}$$

where we have used the matrix inversion lemma and we have defined $\mathbf{B} = (\mathbf{I} + \tilde{\mathbf{T}}^{1/2} \mathbf{K} \tilde{\mathbf{T}}^{1/2})$.

Moreover, once one set of approximate parameters is updated the approximate posterior mean and covariance have to be updated as well and this can be done efficiently with Rank-1 updates whose cost is $O(N^2)$

$$\boldsymbol{\Sigma}^{\text{new}} = \boldsymbol{\Sigma}^{\text{old}} - \frac{\tilde{\boldsymbol{t}}_n^{\text{new}} - \tilde{\boldsymbol{t}}_n^{\text{old}}}{1 + (\tilde{\boldsymbol{t}}_n^{\text{new}} - \tilde{\boldsymbol{t}}_n^{\text{old}}) \boldsymbol{\Sigma}_{n,n}^{\text{old}}} \tilde{\boldsymbol{t}}_n \tilde{\boldsymbol{t}}_n^T.$$

In the above equation $\tilde{\boldsymbol{t}}_n$ are columns of $\tilde{\mathbf{T}}$.

For computing the marginal likelihood care must be taken to avoid numerical over(under)-flows arising from arbitrarily small \tilde{t}_n . Therefore, we compute the log of the marginal and also write the local parameters, $\tilde{\sigma}_n^2, \tilde{\mu}_n$, as functions of the marginal approximate parameters, σ_n^2, μ_n and the cavity's moments, σ_{-n}^2, μ_{-n} . The log marginal for the binary EP is calculated in lines 19 and 20 of Algorithm 2.1. For a detailed derivation see Section A.2.

For making predictions we can again write the approximate predictive mean and variance from Eqs. (2.18) and (2.19) using the transformations above to get

$$\begin{aligned} \mathbb{E}_q[f_* | \mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\boldsymbol{\theta}}] &= \mathbf{k}_*^T (\mathbf{K} + \tilde{\mathbf{T}}^{-1})^{-1} \tilde{\mathbf{T}}^{-1} \tilde{\mathbf{v}} = \mathbf{k}_*^T (\mathbf{I} - (\mathbf{K} + \tilde{\mathbf{T}}^{-1})^{-1} \mathbf{K}) \tilde{\mathbf{v}} \\ &= \mathbf{k}_*^T (\mathbf{I} - \tilde{\mathbf{T}}^{1/2} \mathbf{B}^{-1} \tilde{\mathbf{T}}^{1/2} \mathbf{K}) \tilde{\mathbf{v}}, \end{aligned}$$

$$\begin{aligned} \text{Var}_q[f_* | \mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\boldsymbol{\theta}}] &= k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^T (\mathbf{K} + \tilde{\mathbf{T}}^{-1})^{-1} \mathbf{k}_* \\ &= k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^T \tilde{\mathbf{T}}^{1/2} \mathbf{B}^{-1} \tilde{\mathbf{T}}^{1/2} \mathbf{k}_*. \end{aligned}$$

Algorithm 2.2 shows how predictions are made in pseudo-code.

Algorithm 2.2 EP predictions for binary classification

- 1: **procedure** EP_BINARY_PREDICTION($\mathbf{X}, \mathbf{y}, \tilde{\mathbf{v}}, \tilde{\mathbf{t}}, \mathbf{x}_*, k(\cdot, \cdot)$)
 - 2: $\mathbf{L} = \text{Chol}(\mathbf{I} + \tilde{\mathbf{T}}^{1/2} \mathbf{K} \tilde{\mathbf{T}}^{1/2})$
 - 3: $\mathbf{z} = \tilde{\mathbf{T}}^{1/2} \mathbf{L}^T \backslash (\mathbf{L} \backslash (\tilde{\mathbf{T}}^{1/2} \mathbf{K} \tilde{\mathbf{v}}))$
 - 4: $\mathbb{E}_q[f_*] = \mathbf{k}_*^T (\tilde{\mathbf{v}} - \mathbf{z})$
 - 5: $\mathbf{v} = \mathbf{L} \backslash (\tilde{\mathbf{T}}^{1/2} \mathbf{k}_*)$
 - 6: $\text{Var}_q[f_*] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{v}^T \mathbf{v}$
 - 7: $q_{EP}(y_*) = \Phi(\mathbb{E}_q[f_*] / \sqrt{1 + \text{Var}_q[f_*]})$
 - 8: **return** $q_{EP}(y_*)$
-

2.4.2. The multi-class case

For multiple classes we follow the same procedure for deriving an EP algorithm as in the previous section. We note that the exact posterior using the multinomial probit function is now

$$p(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \frac{1}{Z} \mathcal{N}(\mathbf{f}|\mathbf{0}, \mathbf{K}) \prod_{n=1}^N \int \mathcal{N}(u_n|0, 1) \prod_{j=1, j \neq y_n}^C \Phi(u_n + f_n^{y_n} - f_n^j) du_n.$$

In the approximate posterior each likelihood term is approximated by a scaled Gaussian, i.e,

$$q_{EP}(\mathbf{f}|\mathbf{X}, \mathbf{y}, \boldsymbol{\theta}) = \frac{1}{Z_{EP}} p(\mathbf{f}|\mathbf{X}, \boldsymbol{\theta}) \prod_{n=1}^N \tilde{Z}_n \mathcal{N}\left(\mathbf{f}_n | \tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n\right), \quad (2.20)$$

where \tilde{Z}_n , $\tilde{\boldsymbol{\mu}}_n$, and $\tilde{\boldsymbol{\Sigma}}_n$ are the approximation parameters that we have to estimate using EP.

Again we start with obtaining the *cavity* distribution by removing one approximate term from the approximate posterior to obtain

$$\begin{aligned} q_{-n}(\mathbf{f}_n) &= \mathcal{N}(\mathbf{f}_n | \boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n) \mathcal{N}\left(\mathbf{f}_n | \tilde{\boldsymbol{\mu}}_n, \tilde{\boldsymbol{\Sigma}}_n\right)^{-1} \\ &= \mathcal{N}(\mathbf{f}_n | \boldsymbol{\mu}_{-n}, \boldsymbol{\Sigma}_{-n}) \end{aligned}$$

where $\boldsymbol{\Sigma}_{-n} = (\boldsymbol{\Sigma}_n^{-1} - \tilde{\boldsymbol{\Sigma}}_n^{-1})^{-1}$ and $\boldsymbol{\mu}_{-n} = \boldsymbol{\Sigma}_{-n}(\boldsymbol{\Sigma}_n^{-1} \boldsymbol{\mu}_n - \tilde{\boldsymbol{\Sigma}}_n^{-1} \tilde{\boldsymbol{\mu}}_n)$. We then need to find the *tilted* distribution by matching its moments with the moments of the *cavity* multiplied with the exact term, i.e.

$$\begin{aligned} \hat{q}(\mathbf{f}_n) &= \hat{Z}_n \mathcal{N}\left(\mathbf{f}_n | \hat{\boldsymbol{\mu}}_n, \hat{\boldsymbol{\Sigma}}_n\right) \\ &\approx q_{-n}(\mathbf{f}_n) \int \mathcal{N}(u_n|0, 1) \prod_{j=1, j \neq y_n}^C \Phi(u_n + f_n^{y_n} - f_n^j) du_n. \quad (2.21) \end{aligned}$$

Unlike the binary probit case, where the *tilted* distribution (2.13) is univariate and thus its moments are easy to compute, the *tilted* distribution for the multinomial probit model is C -dimensional. Previous work on EP approximations for the multinomial probit model [8] further approximated the moments of the *tilted* distribution using the Laplace approximation. This assumes that the distributions can be closely approximated by a multivariate normal. [9] use C dimensional numerical integration which for large values of C is problematic. [10] replace the multinomial probit function with a threshold function which results in an EP algorithm similar to the EP for the binary problem. However the algorithm scales as $O((C-1)^3 N^3)$ which again for large C it becomes prohibitively expensive.

The recent work in [5] proposed to approximate the *tilted* distribution also with EP. They call their method nested EP since an inner EP algorithm is used at each iteration of the outer EP algorithm to approximate the moments of the *tilted* distribution. However, they show that the algorithm can be seen as a standard EP with $N(C - 1)$ approximate terms and thus the inner EP only requires a single iteration provided that the approximate parameters are not initialised and they are started from their previous values. Furthermore they show that the structure of the site precision matrices $\tilde{\Sigma}_n^{-1}$ is such that an efficient implementation which scales as $O((C - 1) * N^3)$ is possible. This is similar to the computational complexity of the Laplace approximation even though EP gives a better approximation to the posterior.

2.4.2.1. EP approximation of the tilted distribution

To approximate the *tilted* distribution first an augmentation is used. The vectors f_n are augmented with the corresponding u_n variables to form $w_n = [f_n^T, u_n]^T$. Then ignoring the integration over the auxiliary variable u_n the *tilted* distribution can be written as

$$\hat{p}(w_n) = \hat{Z}_n^{-1} \mathcal{N}(w_n | \mu_{w_n}, \Sigma_{w_n}) \prod_{j=1, j \neq y_n}^C \Phi(w_n^T \tilde{b}_{n,j}) \quad (2.22)$$

where $\mu_{w_n} = [\mu_{-n}^T, 0]^T$ and Σ_{w_n} is a block diagonal matrix with blocks Σ_{-n} and 1. The vectors $\tilde{b}_{n,j}$ are such that $w_n^T \tilde{b}_{n,j} = u_n + f_n^{y_n} - f_n^j$. That is, $\tilde{b}_{n,j} = [(e_{y_n} - e_j)^T, 1]$ where e_j is a C dimensional vector with all elements set to zero and the j -th element set to one. Notice that the moments of (2.22) are equal to the moments of (2.21) since if we integrate over w_n we implicitly integrate over u_n . Also notice that now (2.22) is similar to the posterior for the binary case with $C - 1$ terms which will be approximated by scaled Gaussians.

In [5] (2.22) is approximated by

$$q(w_n) = Z_{\hat{q}_n}^{-1} \mathcal{N}(w_n | \mu_{w_n}, \Sigma_{w_n}) \prod_{j=1, j \neq y_n}^C \tilde{Z}_{n,j} \mathcal{N}\left(w_n^T \tilde{b}_{n,j} | \tilde{\alpha}_{n,j}^{-1} \tilde{\beta}_{n,j}, \tilde{\alpha}_{n,j}^{-1}\right). \quad (2.23)$$

Defining the matrices $\tilde{B}_n = [\tilde{b}_{n,j}]_{j \neq y_n}$, and $\tilde{T}_n = \text{diag}(\tilde{\alpha})$ we can write the approximate distribution as

$$q(w_n) = \mathcal{N}(w_n | \mu_{\hat{q}_n}, \Sigma_{\hat{q}_n})$$

where $\Sigma_{\hat{q}_n} = (\Sigma_{w_n}^{-1} + \tilde{B}_n \tilde{T}_n \tilde{B}_n^T)^{-1}$ and $\mu_{\hat{q}_n} = \Sigma_{\hat{q}_n}^{-1} (\Sigma_{w_n}^{-1} \mu_{w_n} + \tilde{B}_n \tilde{\beta}_n)$.

We once more follow the standard EP procedure and form the *cavity* distribution. However, notice that the approximate terms depend on linear combinations of \mathbf{w}_n . Thus we need to use the marginal approximate distribution of $\mathbf{w}_n^T \tilde{\mathbf{b}}_{n,j}$ which has mean $\mu_{n,j} = \tilde{\mathbf{b}}_{n,j}^T \boldsymbol{\mu}_{\hat{q}_n}$ and variance $\sigma_{n,j}^2 = \tilde{\mathbf{b}}_{n,j}^T \boldsymbol{\Sigma}_{\hat{q}_n} \tilde{\mathbf{b}}_{n,j}$. The *cavity* can then be written as

$$\begin{aligned} q_{-n,j}(\mathbf{w}_n^T \tilde{\mathbf{b}}_{n,j}) &= \mathcal{N}\left(\mathbf{w}_n^T \tilde{\mathbf{b}}_{n,j} | \mu_{n,j}, \sigma_{n,j}^2\right) \mathcal{N}\left(\mathbf{w}_n^T \tilde{\mathbf{b}}_{n,j} | \tilde{\alpha}_{n,j}^{-1} \tilde{\beta}_{n,j}, \tilde{\alpha}_{n,j}^{-1}\right)^{-1} \\ &= \mathcal{N}\left(\mathbf{w}_n^T \tilde{\mathbf{b}}_{n,j} | \mu_{-n,j}, \sigma_{-n,j}^2\right) \end{aligned}$$

where $\sigma_{-n,j}^2 = (\sigma_{n,j}^2 - \tilde{\alpha}_{n,j})$ and $\mu_{-n,j} = \sigma_{-n,j}^2 (\sigma_{n,j}^{-2} m_{n,j} - \tilde{\beta}_{n,j})$.

The corresponding *tilted* distribution for the inner EP is then

$$\begin{aligned} \hat{q}(\mathbf{w}_n^T \tilde{\mathbf{b}}_{n,j}) &= \hat{Z}_{n,j}^{-1} \mathcal{N}\left(\mathbf{w}_n^T \tilde{\mathbf{b}}_{n,j} | \hat{m}_{n,j}, \hat{v}_{n,j}\right) \\ &\approx \mathcal{N}\left(\mathbf{w}_n^T \tilde{\mathbf{b}}_{n,j} | \mu_{-n,j}, \sigma_{-n,j}^2\right) \Phi(\mathbf{w}_n^T \tilde{\mathbf{b}}_{n,j}). \end{aligned}$$

Notice the similarity with (2.13). Both are the product of a univariate Gaussian and a probit and thus the moments can also be analytically derived as

$$\hat{Z}_{n,j} = \Phi(z_{n,j}), \quad (2.24)$$

$$\hat{\mu}_{n,j} = \mu_{-n,j} + \frac{\sigma_{-n,j}^2 \mathcal{N}(z_{n,j})}{\Phi(z_{n,j}) \sqrt{1 + \sigma_{-n,j}^2}}, \quad (2.25)$$

$$\hat{\sigma}_{n,j}^2 = \sigma_{-n,j}^2 - \frac{\sigma_{-n,j}^4 \mathcal{N}(z_{n,j})}{(1 + \sigma_{-n,j}^2 \Phi(z_{n,j}))} \left(z_{n,j} + \frac{\mathcal{N}(z_{n,j})}{\Phi(z_{n,j})} \right), \quad (2.26)$$

$$z_{n,j} = \frac{\mu_{-n,j}}{\sqrt{1 + \sigma_{-n,j}^2}}. \quad (2.27)$$

The local approximation parameters are then be updated in a similar fashion as in the binary case.

$$\begin{aligned} \Delta \tilde{\alpha}_{n,j} &= \hat{\sigma}_{n,j}^{-2} - \sigma_{-n,j}^{-2} - \tilde{\alpha}_{n,j}, \\ \Delta \tilde{\beta}_{n,j} &= \hat{\sigma}_{n,j}^{-2} \hat{\mu}_{n,j} - \sigma_{-n,j}^{-2} \mu_{-n,j} - \tilde{\beta}_{n,j}, \\ \tilde{\alpha}_{n,j} &= \tilde{\alpha}_{n,j} + \Delta \tilde{\alpha}_{n,j}, \\ \tilde{\beta}_{n,j} &= \tilde{\beta}_{n,j} + \Delta \tilde{\beta}_{n,j}, \\ \tilde{Z}_{n,j} &= \hat{Z}_{n,j} \mathcal{N}\left(\mu_{-n,j} | \tilde{\alpha}_{n,j}^{-1} \tilde{\beta}_{n,j}, \sigma_{-n,j}^2 + \tilde{\alpha}_{n,j}^{-1}\right)^{-1} \end{aligned}$$

Once the approximate parameters for the j -th term have been updated, the mean and covariance of the approximate *tilted* distribution need to also be updated before moving to the next approximate term. This can be done with Rank-1 updates as we did in the binary case. A detailed implementation can be found in the next section of this chapter.

Once a single pass over all the j approximate terms is done the normalisation term needs to be computed. Since the approximate *tilted* (2.23) is a product of Gaussians its normalisation term can also be obtained in a similar manner as in the binary case, that is

$$q(\mathbf{w}_n) = Z_{\hat{q}_n}^{-1} \mathcal{N}(\mathbf{w}_n | \boldsymbol{\mu}_{\hat{q}_n}, \boldsymbol{\Sigma}_{\hat{q}_n}) \mathcal{N}\left(\boldsymbol{\mu}_{w_n} | \tilde{\mathbf{B}}_n \tilde{\boldsymbol{\beta}}_n, \boldsymbol{\Sigma}_{\hat{q}_n} + \tilde{\mathbf{B}}_n \tilde{\mathbf{T}}_n \tilde{\mathbf{B}}_n^T\right) \times \\ \prod_{j=1, j \neq y_n}^C \tilde{Z}_{n,j},$$

$$Z_{\hat{q}_n} = \mathcal{N}\left(\boldsymbol{\mu}_{w_n} | \tilde{\mathbf{B}}_n \tilde{\boldsymbol{\beta}}_n, \boldsymbol{\Sigma}_{\hat{q}_n} + \tilde{\mathbf{B}}_n \tilde{\mathbf{T}}_n \tilde{\mathbf{B}}_n^T\right) \prod_{j=1, j \neq y_n}^C \tilde{Z}_{n,j}.$$

We now have approximated the moments of the *tilted* distribution (2.22) using EP. Before proceeding to the next iteration of the outer EP we first need to update the approximate covariance matrix $\hat{\boldsymbol{\Sigma}}_n$ and mean $\hat{\boldsymbol{\mu}}_n$. In Equation (2.22) we defined the augmented cavity for \mathbf{w}_n by augmenting \mathbf{f}_n and u_n and neglecting the integration over u_n . In order to obtain the approximate *tilted* moments for the latent variables \mathbf{f}_n we simply have to take the marginal of the approximate distribution and thus extract the corresponding terms from the covariance $\hat{\boldsymbol{\Sigma}}_{\hat{q}_n}$ and mean $\boldsymbol{\mu}_{\hat{q}_n}$. If we define the matrix $\mathbf{H}^T = [\mathbf{I}_C \quad \mathbf{0}]$ then the approximate marginal covariance and mean of the *tilted* distribution for \mathbf{f}_n are

$$\hat{\boldsymbol{\Sigma}}_n = \mathbf{H}^T \boldsymbol{\Sigma}_{\hat{q}_n} \mathbf{H} = \mathbf{H}^T (\boldsymbol{\Sigma}_{w_n}^{-1} + \tilde{\mathbf{B}}_n \tilde{\mathbf{T}}_n \tilde{\mathbf{B}}_n^T)^{-1} \mathbf{H}$$

$$\hat{\boldsymbol{\mu}}_n = \mathbf{H}^T \boldsymbol{\mu}_{\hat{q}_n} = \mathbf{H}^T (\boldsymbol{\Sigma}_{w_n}^{-1} + \tilde{\mathbf{B}}_n \tilde{\mathbf{T}}_n \tilde{\mathbf{B}}_n^T)^{-1} (\boldsymbol{\Sigma}_{w_n}^{-1} \boldsymbol{\mu}_{w_n} + \tilde{\mathbf{B}}_n \tilde{\boldsymbol{\beta}}_n).$$

We can then update the approximate parameters such that they have the desired moments

$$\tilde{\boldsymbol{\Sigma}}_n = (\hat{\boldsymbol{\Sigma}}_n^{-1} - \boldsymbol{\Sigma}_{-n}^{-1})^{-1}$$

$$\tilde{\boldsymbol{\mu}}_n = \tilde{\boldsymbol{\Sigma}}_n (\hat{\boldsymbol{\Sigma}}_n^{-1} \hat{\boldsymbol{\mu}}_n - \boldsymbol{\Sigma}_{-n} \boldsymbol{\mu}_{-n})$$

$$\tilde{Z}_n = Z_{\hat{q}_n} \mathcal{N}\left(\boldsymbol{\mu}_{-n} | \tilde{\boldsymbol{\mu}}_n, \boldsymbol{\Sigma}_{-n} + \tilde{\boldsymbol{\Sigma}}_n\right)^{-1}.$$

The approximate posterior obtained by the EP algorithm is then also a multivariate normal distribution $\mathcal{N}(\mathbf{f}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with mean and covariance

$$\boldsymbol{\Sigma} = (\mathbf{K}^{-1} + \tilde{\boldsymbol{\Sigma}}^{-1})^{-1},$$

$$\boldsymbol{\mu} = \boldsymbol{\Sigma} \tilde{\boldsymbol{\Sigma}}^{-1} \tilde{\boldsymbol{\mu}},$$

where $\tilde{\boldsymbol{\mu}}$ is obtained by stacking the vectors $\tilde{\boldsymbol{\mu}}_n$ and the elements of $\tilde{\boldsymbol{\Sigma}}$ are taken from the matrices $\tilde{\boldsymbol{\Sigma}}_n$. Notice that the form of $\tilde{\boldsymbol{\Sigma}}$ is not specified till now but we will describe it in more detail in the implementation section for clarity. The normalisation term of the approximate posterior again follows from the properties of the normal distribution since the approximate posterior from Equation (2.20) can be written as a product of two multivariate Normals, thus

$$Z_{EP} = \mathcal{N}\left(\tilde{\boldsymbol{\mu}}|\mathbf{0}, \mathbf{K} + \tilde{\boldsymbol{\Sigma}}\right) \prod_{n=1}^N \tilde{Z}_n.$$

Similarly the predictive mean and covariance for new observations have the same forms with those presented for the binary case in Eqs. (2.18) and (2.19). Finally, to obtain the predictive class probabilities we need to integrate over the *latent* \mathbf{f}_* however this is equivalent to computing the moment of the *titled* distribution $Z_{\hat{q}_n}$. For more details see the implementation section below.

2.4.2.2. Implementation

In this subsection we present an efficient and practical implementation for the EP algorithm for multi-class classification problems which scales as $O((C-1)N^3)$. We follow [5] to show that the precision matrices and means of the approximate terms depend only on $\tilde{\alpha}_{n,j}$ and $\tilde{\beta}_{n,j}$. The matrices of central importance are the site precisions $\tilde{\boldsymbol{\Sigma}}_n^{-1}$, as well as the approximate covariance matrix $\boldsymbol{\Sigma}$ and the matrix with all site precisions $\tilde{\boldsymbol{\Sigma}}^{-1}$. The aim is to avoid inverting directly the $CN \times CN$ matrix $\boldsymbol{\Sigma}$ and exploit the block diagonal structure of the prior covariance matrix \mathbf{K} and the sparse structure of $\tilde{\boldsymbol{\Sigma}}^{-1}$. Similar to the EP for binary classification we try to compute the log marginals of the *tilted* and approximate distributions by avoiding using the site locations and site precision matrices directly to avoid numerical instabilities.

The site precision parameters can be written as, see Section A.3 in the Appendix for a detailed derivation or [5],

$$\tilde{\boldsymbol{\Sigma}}_n^{-1} = \text{diag}(\boldsymbol{\pi}_n) - (\mathbf{1}_C^T \boldsymbol{\pi}_n)^{-1} \boldsymbol{\pi}_n \boldsymbol{\pi}_n^T, \quad (2.28)$$

where $\mathbf{E}_{-y_n} = [\mathbf{e}_j]_{j \neq y_n}$ and $\boldsymbol{\pi}_n = \mathbf{E}_{-y_n} \tilde{\boldsymbol{\alpha}}_n - \mathbf{e}_{y_n}$. Analogously we can now define the matrix with all site precisions as

$$\tilde{\boldsymbol{\Sigma}}^{-1} = \mathbf{D} - \mathbf{D}\mathbf{R}(\mathbf{R}^T\mathbf{D}\mathbf{R})^{-1}\mathbf{R}^T\mathbf{D},$$

where $\mathbf{D} = \text{diag}([\pi_1^1, \dots, \pi_n^1, \pi_1^2, \dots, \pi_n^2, \dots, \pi_1^C, \dots, \pi_n^C]^T)$ and \mathbf{R} is a $CN \times N$ matrix with C identity matrices \mathbf{I}_N stacked together. Also the approximate location parameters, $\tilde{\mathbf{v}}_n = \tilde{\boldsymbol{\Sigma}}_n^{-1} \tilde{\boldsymbol{\mu}}_n$ can also be written as, again the details of the derivation are provided in Section A.3 of the Appendix or [5],

$$\tilde{\mathbf{v}}_n = (\mathbf{1}^T \tilde{\boldsymbol{\beta}}_n)(\mathbf{1}^T \boldsymbol{\pi}_n)^{-1} \boldsymbol{\pi}_n - \mathbf{E}_{-y_n} \tilde{\boldsymbol{\beta}}_n. \quad (2.29)$$

We can see that both Equations (2.28) and (2.29) only depend on the parameters of the *tilted* approximation, $\tilde{\alpha}_{n,j}$ and $\tilde{\beta}_{n,j}$ which greatly simplifies the implementation as we only have to keep track of only those parameters.

The covariance of the approximate distribution is then $\boldsymbol{\Sigma} = (\mathbf{K}^{-1} + \tilde{\boldsymbol{\Sigma}}^{-1})^{-1}$. Naively computing this matrix will result in an algorithm that scales as $O(C^3 N^3)$. By using the matrix inversion lemma we can write it as $\boldsymbol{\Sigma} = \mathbf{K} - \mathbf{K}\mathbf{M}\mathbf{K}^T$, where we the matrix \mathbf{M} is

$$\mathbf{M} = \mathbf{B} - \mathbf{B}\mathbf{R}(\mathbf{R}^T\mathbf{B}\mathbf{R})^{-1}\mathbf{R}^T\mathbf{B}$$

and $\mathbf{B} = \mathbf{D}^{1/2}(\mathbf{D}^{1/2}\mathbf{K}\mathbf{D}^{1/2} + \mathbf{I})^{-1}\mathbf{D}^{1/2}$. For a detailed derivation see Section A.4 in the Appendix. Given that \mathbf{K} is block diagonal by definition then \mathbf{B} is also block diagonal. Also $\mathbf{R}^T\mathbf{B}\mathbf{R}$ is a matrix of size $N \times N$ and is equal to $\sum_{j=1}^C \mathbf{B}_j$, i.e. the sum of the blocks of \mathbf{B} . Also $\mathbf{R}\mathbf{A}\mathbf{R}^T$ is a $CN \times CN$ block matrix with all blocks equal to \mathbf{A} .

Using the above representation for the approximate covariance we can also get an efficient representation for computing the approximate mean, $\boldsymbol{\mu} = \boldsymbol{\Sigma}\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\boldsymbol{\mu}}$, by using the site location parameters $\tilde{\mathbf{v}}$ to get

$$\boldsymbol{\mu} = \mathbf{K}\tilde{\mathbf{v}} - \mathbf{K}\mathbf{M}\mathbf{K}\tilde{\mathbf{v}}.$$

Algorithm 2.3 approximates the inner *tilted* distribution in Equation (2.22) and Algorithm 2.4 is the EP for the approximation of the multinomial probit and depends on Algorithm 2.3. Similar to the binary EP we use the following transformations

$$\mathbf{T}_{w_n} = \boldsymbol{\Sigma}_{w_n}^{-1}, \quad \mathbf{v}_{w_n} = \mathbf{T}_{w_n} \boldsymbol{\mu}_{w_n}, \quad \mathbf{T}_{-n} = \boldsymbol{\Sigma}_{-n}^{-1}, \quad \mathbf{v}_{-n} = \mathbf{T}_{-n} \boldsymbol{\mu}_{-n}.$$

Notice that the expensive calculations in lines 17–22 of Algorithm 2.3 they are only needed for computing the normalisation constant of the *tilted* which, in Algorithm 2.4 is only required after convergence. Thus in an actual implementation they should only be run once after convergence of the outer EP.

Algorithm 2.3 EP approximation of the *tilted* distribution

```

1: procedure EP_TILTED( $\mathbf{v}_{w_n}, \mathbf{T}_{w_n}, \tilde{\boldsymbol{\alpha}}_n, \tilde{\boldsymbol{\beta}}_n, y_n$ )
2:    $\Sigma_{\hat{q}_n} = (\mathbf{T}_{w_n} + \tilde{\mathbf{B}}_n \tilde{\mathbf{T}}_n \tilde{\mathbf{B}}_n^T)^{-1}$ ,    $\boldsymbol{\mu}_{\hat{q}_n} = \Sigma_{\hat{q}_n}(\mathbf{v}_{w_n} + \tilde{\mathbf{B}}_n \tilde{\boldsymbol{\beta}}_n)$ 
3:   for  $j := 1$  to  $C$ ,  $j \neq y_n$  do
4:      $\sigma_{n,j}^2 = \tilde{\mathbf{b}}_{n,j}^T \Sigma_{\hat{q}_n} \tilde{\mathbf{b}}_{n,j}$ ,    $\mu_{n,j} = \tilde{\mathbf{b}}_{n,j} \boldsymbol{\mu}_{\hat{q}_n}$        $\triangleright$  Marginal parameters
5:      $\sigma_{-n,j}^2 = (\sigma_{n,j}^{-2} - \tilde{\alpha}_{n,j})^{-1}$            $\triangleright$  Compute inner cavity
6:      $\mu_{-n,j} = \sigma_{-n,j}^2 (\sigma_{n,j}^{-2} \mu_{n,j} - \tilde{\beta}_{n,j})$ 
7:     Compute moments of inner tilted,  $\hat{Z}_{n,j}, \hat{\mu}_{n,j}, \hat{\sigma}_{n,j}^2$  using (2.24–
     2.27).
8:      $\Delta \tilde{\alpha}_{n,j} = \hat{\sigma}_{n,j}^{-2} - \sigma_{-n,j}^{-2} - \tilde{\alpha}_{n,j}$ 
9:      $\Delta \tilde{\beta}_{n,j} = \hat{\sigma}_{n,j}^{-2} \hat{\mu}_{n,j} - \sigma_{-n,j}^{-2} \mu_{-n,j} - \tilde{\beta}_{n,j}$ 
10:     $\tilde{\alpha}_{n,j} = \tilde{\alpha}_{n,j} + \Delta \tilde{\alpha}_{n,j}$                        $\triangleright$  Update Parameters
11:     $\beta_{n,j} = \beta_{n,j} - \Delta \tilde{\beta}_{n,j}$ 
12:     $\triangleright$  Update tilted approximation
13:     $\Sigma_{\hat{q}_n} = \Sigma_{\hat{q}_n} + \Sigma_{\hat{q}_n} \tilde{\mathbf{b}}_{n,j} (1 + \Delta \tilde{\alpha}_{n,j} \sigma_{n,j}^2)^{-1} \Delta \tilde{\alpha}_{n,j} \tilde{\mathbf{b}}_{n,j}^T \Sigma_{\hat{q}_n}$ 
14:     $\boldsymbol{\mu}_{\hat{q}_n} = \boldsymbol{\mu}_{\hat{q}_n} + \Sigma_{\hat{q}_n} \tilde{\mathbf{b}}_{n,j} (1 + \Delta \tilde{\alpha}_{n,j} \sigma_{n,j}^2)^{-1} (\Delta \tilde{\beta}_{n,j} - \Delta \tilde{\alpha}_{n,j} \mu_{n,j})$ 
15:  end for
16:   $\triangleright$  Compute log of the normalisation constant
17:   $\mathbf{L}_{w_n} = \text{chol}(\mathbf{T}_{w_n})$ ,    $\boldsymbol{\mu}_{w_n} = \mathbf{L}_{w_n}^T \setminus (\mathbf{L}_{w_n} \setminus \mathbf{v}_{w_n})$ 
18:   $\mathbf{L}_{\hat{q}_n} = \text{chol}(\Sigma_{\hat{q}_n})$ ,    $\mathbf{v}_{\hat{q}_n} = \mathbf{L}_{\hat{q}_n}^T \setminus (\mathbf{L}_{\hat{q}_n} \setminus \boldsymbol{\mu}_{\hat{q}_n})$ 
19:   $\log |\Sigma_{\hat{q}_n}| = \sum_{j=1}^C \log (\mathbf{L}_{\hat{q}_n})_{j,j}$ ,    $\log |\Sigma_{w_n}| = - \sum_{j=1}^C \log (\mathbf{L}_{w_n})_{j,j}$ 
20:   $\log Z_{\hat{q}_n} = \frac{1}{2} \boldsymbol{\mu}_{\hat{q}_n}^T \mathbf{v}_{\hat{q}_n} + \frac{1}{2} \log |\Sigma_{\hat{q}_n}| - \frac{1}{2} \boldsymbol{\mu}_{w_n}^T \mathbf{v}_{w_n} - \frac{1}{2} \log |\Sigma_{w_n}|$ 
21:   $+ \sum_{j=1, j \neq y_n}^C \log \hat{Z}_{n,j} + \frac{1}{2} \sum_{j=1, j \neq y_n}^C (\mu_{-n,j}^2 \sigma_{-n,j}^{-2} + \log \sigma_{-n,j}^2)$ 
22:   $+ \frac{1}{2} \sum_{j=1, j \neq y_n}^C (\mu_{n,j}^2 \sigma_{n,j}^{-2} + \log \sigma_{n,j}^2)$ 
23: return  $\tilde{\boldsymbol{\alpha}}_n, \tilde{\boldsymbol{\beta}}_n, \log Z_{\hat{q}_n}$ 

```

Algorithm 2.4 EP approximation for the multinomial probit model

```

1: procedure EP_MULTINOMIAL( $\mathbf{K}, \mathbf{y}$ )
2:    $\tilde{\boldsymbol{\alpha}} = \mathbf{0}$ ,  $\tilde{\boldsymbol{\beta}} = \mathbf{0}$ ,  $\boldsymbol{\Sigma} = \mathbf{K}$ ,  $\boldsymbol{\mu} = \mathbf{0}$ 
3:   repeat
4:     for  $n := 1$  to  $N$  do
5:        $\mathbf{T}_{-n} = (\boldsymbol{\Sigma}_n^{-1} - \tilde{\boldsymbol{\Sigma}}_n^{-1})$ ,  $\mathbf{v}_{-n} = (\boldsymbol{\Sigma}_n^{-1} \boldsymbol{\mu}_n - \tilde{\boldsymbol{\Sigma}}_n^{-1} \tilde{\boldsymbol{\mu}}_n)$ 
6:        $\mathbf{T}_{w_n} = \text{blockdiag}(\mathbf{T}_{-n}, 1)$ ,  $\mathbf{v}_{w_n} = [\mathbf{v}_{-n} \mathbf{0}]^T$ 
7:        $[\tilde{\boldsymbol{\alpha}}_n, \tilde{\boldsymbol{\beta}}_n, \log Z_{\hat{q}_n}] = \text{EP\_tilted}(\mathbf{v}_{w_n}, \mathbf{T}_{w_n}, \tilde{\boldsymbol{\alpha}}_n, \tilde{\boldsymbol{\beta}}_n, y_n)$ 
8:        $\boldsymbol{\pi}_n = E_{-\mathbf{y}_n} \tilde{\boldsymbol{\alpha}}_n + \mathbf{e}_{y_n}$ 
9:        $\tilde{\boldsymbol{\Sigma}}_n^{-1} = \text{diag}(\boldsymbol{\pi}_n) - (\mathbf{1}_c^T \boldsymbol{\pi}_n)^{-1} \boldsymbol{\pi}_n \boldsymbol{\pi}_n^T$   $\triangleright$  Update parameters
10:       $\tilde{\mathbf{v}}_n = (\mathbf{1}^T \tilde{\boldsymbol{\beta}}_n) / (\mathbf{1}^T \boldsymbol{\pi}_n) - E_{-\mathbf{y}_n} \tilde{\boldsymbol{\beta}}_n$ 
11:    end for
12:    for  $j := 1$  to  $C$  do  $\triangleright$  Update approx. mean and covariance
13:       $\mathbf{A}_j = \mathbf{D}_j^{1/2} \mathbf{K}_j \mathbf{D}_j^{1/2} + \mathbf{I}$ ,  $\mathbf{L}_j = \text{chol}(\mathbf{A}_j)$ 
14:       $\mathbf{L}\mathbf{D}_j = \mathbf{L}_j \setminus \mathbf{D}_j^{1/2}$ ,  $\mathbf{B}_j = \mathbf{L}\mathbf{D}_j^T \mathbf{L}\mathbf{D}_j$ ,
15:       $\mathbf{B}\mathbf{K}_j = \mathbf{B}_j \mathbf{K}_j$ ,  $\mathbf{B}\mathbf{K}\mathbf{v}_j = \mathbf{B}\mathbf{K}_j \tilde{\mathbf{v}}_j$ 
16:    end for
17:     $\mathbf{P} = \sum_{j=1}^C \mathbf{B}_j$ ,  $\mathbf{L}_p = \text{chol}(\mathbf{P})$ 
18:    for  $j := 1$  to  $C$  do
19:       $\mathbf{L}\mathbf{B}\mathbf{K}_j = \mathbf{L}_p \setminus \mathbf{B}\mathbf{K}_j^T$ ,  $\mathbf{L}\mathbf{L}\mathbf{B}\mathbf{K}\mathbf{v} = \mathbf{L}_p^T \setminus (\mathbf{L}_p \setminus \sum_{j=1}^C \mathbf{B}\mathbf{K}\mathbf{v}_j)$ 
20:       $\boldsymbol{\mu}_j = \mathbf{K}_j \tilde{\mathbf{v}}_j - \mathbf{K}_j \mathbf{B}\mathbf{K}\mathbf{v}_j + \mathbf{B}\mathbf{K}_j^T \mathbf{L}\mathbf{L}\mathbf{B}\mathbf{K}\mathbf{v}$ 
21:       $\boldsymbol{\Sigma}_{j,j} = \mathbf{K}_j - \mathbf{K}_j \mathbf{B}\mathbf{K}_j + \mathbf{L}\mathbf{B}\mathbf{K}_j^T \mathbf{L}\mathbf{B}\mathbf{K}_j$ 
22:      for  $k := j + 1$  to  $C$  do
23:         $\boldsymbol{\Sigma}_{j,k} = \boldsymbol{\Sigma}_{k,j} = \mathbf{L}\mathbf{B}\mathbf{K}_j^T \mathbf{L}\mathbf{B}\mathbf{K}_j$ 
24:      end for
25:    end for
26:  until convergence
27:  for  $n := 1$  to  $N$  do  $\triangleright$  Compute log of the normalisation constant
28:     $\mathbf{L}_{-n} = \text{chol}(\mathbf{T}_{-n})$ ,  $\boldsymbol{\mu}_{-n} = \mathbf{L}_{-n}^T \setminus (\mathbf{L}_{-n} \setminus \mathbf{v}_{-n})$ 
29:     $\mathbf{L}_n = \text{chol}(\boldsymbol{\Sigma}_n)$ ,  $\mathbf{v}_n = \mathbf{L}_n^T \setminus (\mathbf{L}_n \setminus \boldsymbol{\mu}_n)$ 
30:     $\log |\mathbf{T}_{-n}| = \sum_{j=1}^C \log (\mathbf{L}_{-n})_{j,j}$ ,  $\log |\boldsymbol{\Sigma}_n| = \sum_{j=1}^C \log (\mathbf{L}_n)_{j,j}$ 
31:  end for
32:   $\log |\mathbf{I} + \mathbf{K}\tilde{\boldsymbol{\Sigma}}^{-1}| = \sum_{j=1}^C \sum_{n=1}^N \log (\mathbf{L}_j)_{n,n} + \sum_{n=1}^N \log (\mathbf{L}_p)_{n,n}$ 
33:           $- \sum_{n=1}^N \log (\sum_{j=1}^C \mathbf{D}_j)_{n,n}$ 
34:   $\log Z_{EP} = \frac{1}{2} \boldsymbol{\mu}^T \tilde{\mathbf{v}} - \frac{1}{2} \log |\mathbf{I} + \mathbf{K}\tilde{\boldsymbol{\Sigma}}^{-1}| + \frac{1}{2} \sum_{n=1}^N (\boldsymbol{\mu}_{-n}^T \mathbf{v}_{-n} - \log |\mathbf{T}_{-n}|)$ 
35:           $- \frac{1}{2} \sum_{n=1}^N (\boldsymbol{\mu}_n^T \mathbf{v}_n + \log |\boldsymbol{\Sigma}_n|) + \sum_{n=1}^N \log Z_{\hat{q}_n}$ 
36:  return  $\boldsymbol{\mu}, \boldsymbol{\Sigma}, \tilde{\mathbf{v}}, \mathbf{L}\mathbf{L}\mathbf{B}\mathbf{K}\mathbf{v}, \mathbf{B}, \mathbf{L}_p, \mathbf{B}\mathbf{K}\mathbf{v}_j, \log Z_{EP}$ 

```

For making predictions we have to first calculate the predictive mean and covariance for the new latent variables \mathbf{f}_* . Similar to the binary case, using the results in Appendix A.1 and that $\mathbf{M} = (\mathbf{K} + \tilde{\boldsymbol{\Sigma}})^{-1}$ we can get

$$\mathbb{E}_q[\mathbf{f}_* | \mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\boldsymbol{\theta}}] = \mathbf{k}_* \tilde{\mathbf{v}} - \mathbf{k}_* \mathbf{M} \mathbf{K} \tilde{\mathbf{v}} \quad (2.30)$$

$$\text{Var}_q[\mathbf{f}_* | \mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\boldsymbol{\theta}}] = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_* \mathbf{M} \mathbf{k}_*^T, \quad (2.31)$$

where \mathbf{k}_* is a $C \times CN$ matrix with the covariance terms between the test point \mathbf{x}_* and the training points \mathbf{X} . For computing the predictive class probabilities we have to integrate over \mathbf{f}_* and as we already noted this is similar of computing the moments of the augment *tilted* distribution. In Algorithm 2.5 we reuse Algorithm 2.3 only this time we initialise the site parameters to 0 and we rerun the algorithm until convergence. The normalising constant is the EP approximation to the integral of the predictive class probabilities.

Algorithm 2.5 EP predictions for the multinomial probit model

```

1: procedure
    EP_MULT_PRED( $\mathbf{K}, \mathbf{y}, \mathbf{k}_*, k(\mathbf{x}_*, \mathbf{x}_*), \tilde{\mathbf{v}}, \mathbf{L} \mathbf{L} \mathbf{B} \mathbf{K} \mathbf{v}, \mathbf{B}, \mathbf{L}_p, \mathbf{B} \mathbf{K} \mathbf{v}_j$ )
2:   for  $j := 1$  to  $C$  do
3:      $\mu_{*j} = \mathbf{k}_{*j} \tilde{\mathbf{v}}_j - \mathbf{k}_{*j} \mathbf{B} \mathbf{K} \mathbf{v}_j + \mathbf{k}_{*j} \mathbf{B}_j \mathbf{L} \mathbf{L} \mathbf{B} \mathbf{K} \mathbf{v}$ 
4:      $\mathbf{L} \mathbf{B} \mathbf{k}_{*j} = \mathbf{L}_p \setminus (\mathbf{B}_j \mathbf{k}_{*j})$ 
5:      $\Sigma_{*(j,j)} = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_{*j} \mathbf{B}_j \mathbf{k}_{*j} + \mathbf{L} \mathbf{B} \mathbf{k}_{*j}^T \mathbf{L} \mathbf{B} \mathbf{k}_{*j}$ 
6:     for  $k := j + 1$  to  $C$  do
7:        $\Sigma_{*(k,j)} = \Sigma_{*(j,k)} = \mathbf{L} \mathbf{B} \mathbf{k}_{*,j}^T \mathbf{L} \mathbf{B} \mathbf{k}_{*,j}$ 
8:     end for
9:   end for
10:   $\mathbf{T} = \text{blkdiag}(\Sigma_*^{-1}, 1), \quad \mathbf{v} = [\boldsymbol{\mu}_*^T \Sigma_*^{-1}, 0]^T$ 
11:  for  $j := 1$  to  $C$  do
12:     $\mathbf{a} = \mathbf{0}, \quad \mathbf{b} = 0$ 
13:    repeat
14:       $[\mathbf{a}, \mathbf{b}, \log Z_{\hat{q}}] = \text{EP\_tilted}(\mathbf{v}, \mathbf{T}, \mathbf{a}, \mathbf{b}, j)$ 
15:    until convergence
16:     $q_{EP}(\mathbf{y}_{*j} = 1) = \exp(\log Z_{\hat{q}})$ 
17:  end for
18: return  $q_{EP}(\mathbf{y}_*)$ 

```

2.5. Application: Bat Call Identification

In this section we present an application of the multinomial probit GP classification on real data. The classification problem is to classify the high frequency sounds that bats use to echolocate into species. The problem is motivated by the need of large scale conservation and ecological studies where one would like to have automated and unattended recording devices left in a region recording bat echolocation calls and then use the recorded data to estimate the population of bat species in the area.

2.5.1. *Dataset*

The data used in this example are collected from North and Central Mexico by capturing bats, identifying its species by expert biologists on site and then recording echolocation calls when a bat is released. More details can be found in [11]. The data consist of 21 species, 449 individual bats (recordings) and 8429 bat echolocation calls. Table 2.1 summarises the data. Care must be taken when splitting the data to training and test sets during cross-validation in order to ensure that calls from the same individual bat recording are not in both sets. For that we split our dataset using recordings instead of calls. For species with less than 100 recordings we include as many calls as possible up to a maximum of 100 calls per species.

2.5.2. *Data representation*

Each call is represented by its spectrogram, Figure 2.3, which is calculated by using a hamming window of size 256 with 95% overlap and an FFT length of 512. The frequency range of the spectrogram is thresholded by removing frequencies below 5kHz and above 210kHz. A vector representation \mathbf{x}_n for each call is constructed by extracting call shape parameters from the call's spectrogram similar to [12]. In total 32 parameters are calculated including the call's duration in miliseconds, the highest and lowest frequencies of the call, its total frequency spread, the frequency with maximum amplitude, the frequencies at the start and end of the call etc.

2.5.3. *Dynamic time warping*

Although extracting call shape parameters from the spectrogram of a call captures some of the call's characteristics and shape, there is still a lot of information that is discarded, e.g. harmonics. An alternative to character-

Table 2.1. Dataset statistics.

Species	Samples	Calls
Family: Emballonuridae		
1 <i>Balantiopteryx plicata</i>	16	384
Family: Molossidae		
2 <i>Nyctinomops femorosaccus</i>	16	311
3 <i>Tadarida brasiliensis</i>	49	580
Family: Mormoopidae		
4 <i>Mormoops megalophylla</i>	10	135
5 <i>Pteronotus davyi</i>	8	106
6 <i>Pteronotus parnellii</i>	23	313
7 <i>Pteronotus personatus</i>	7	51
Family: Phyllostomidae		
8 <i>Artibeus jamaicensis</i>	11	82
9 <i>Desmodus rotundus</i>	6	38
10 <i>Leptonycteris yerbabuenae</i>	26	392
11 <i>Macrotus californicus</i>	6	53
12 <i>Sturnira ludovici</i>	12	71
Family: Vespertilionidae		
13 <i>Antrozous pallidus</i>	58	1937
14 <i>Eptesicus fuscus</i>	74	1589
15 <i>Idionycteris phyllotis</i>	6	177
16 <i>Lasiurus blossevillii</i>	10	90
17 <i>Lasiurus cinereus</i>	5	42
18 <i>Lasiurus xanthinus</i>	8	204
19 <i>Myotis volans</i>	8	140
20 <i>Myotis yumanensis</i>	5	89
21 <i>Pipistrellus hesperus</i>	85	2445

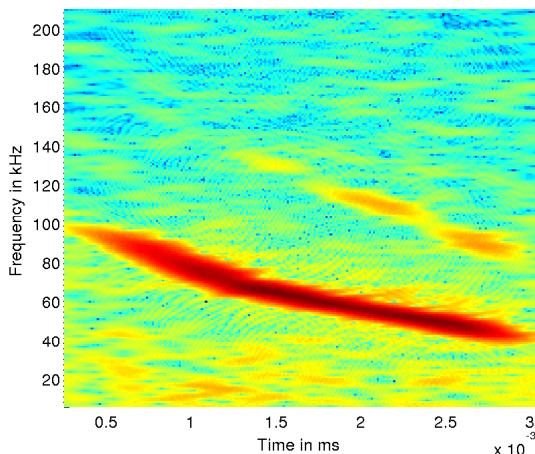


Fig. 2.3. Example of a call's spectrogram. See text for details on spectrogram computation.

ising a call using predefined parameters is to directly utilise its spectrogram. However due to the differences in call duration the spectrograms will need to be normalised in order to have the same length using some form of interpolation. In this work we borrow ideas from speech recognition [13] and previous work on bird call classification [14] and employ the Dynamic Time Warping (DTW) kernel to directly compare two calls' spectrograms.

Given two calls i, j from the library and their spectrograms $\mathbf{S}_i, \mathbf{S}_j$, where $\mathbf{S}_i \in \mathbb{C}^{F \times W}$ with F being the number of frequency bands and W the number of windows, the dissimilarity matrix $\mathbf{D}^{i,j} \in \mathbb{R}^{W \times W}$ is constructed such that

$$\mathbf{D}^{i,j}(w, v) = 1 - \frac{\mathbf{S}_i(:, w)^T \mathbf{S}_j(:, v)}{\sqrt{\mathbf{S}_i(:, w)^T \mathbf{S}_i(:, w) \mathbf{S}_j(:, v)^T \mathbf{S}_j(:, v)}}. \quad (2.32)$$

DTW uses the dissimilarity matrix in order to stretch or expand spectrogram \mathbf{S}_i over time in order to match \mathbf{S}_j by calculating the optimal warping path with the smallest alignment cost, $c_{i,j}$, using dynamic programming. Figure 2.4 illustrates the optimal warping path for two calls in the library.

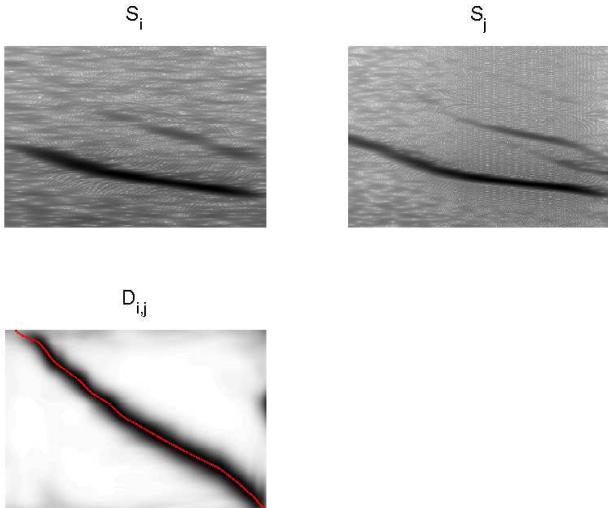


Fig. 2.4. Example of DTW optimal warping path for 2 call spectrograms from the same species. Top row, call spectrograms; bottom row, dissimilarity matrix and optimal warping path.

For each call we construct a vector representation \mathbf{x}_n by computing the optimal warping paths with all N calls from the library and concatenating the alignment costs such that $\mathbf{x}_n = [c_{n,1}, \dots, c_{n,N}]$.

2.5.4. Experiments

We compare the classification accuracy of the multinomial probit regression with Gaussian process prior classifier using the two representations obtained from the spectrogram. For both DTW and call shape parameters we use a squared exponential covariance function. For the the call shape parameters we use independent (ARD) lengthscale parameters. We also combine both covariance functions by a weighted sum where weights are treated as additional parameters.

The values of the call shape parameters are normalised to have zero mean and standard deviation equal to one by subtracting the mean and dividing by the standard deviation of the call shape parameters in the training set. For the 33 covariance function parameters, σ and l_1, \dots, l_{32} we use independent Gamma priors with shape parameter 1.5 and scale parameter 10. For the DTW representation each call vector of optimal alignment costs is normalised to unit length and independent Gamma (1.5, 10) priors are used for the magnitude and length-scale covariance function parameters. The weights for the linear combination of the DTW and call shape kernel functions are restricted to be positive and sum to 1 and a flat Dirichlet prior is used.

As a baseline we also compare with a multi-class Support Vector Machine (SVM) classifier using the LibSVM software library [15]. For the SVM we use the call shape parameters and the DTW representations with a squared exponential kernel. The kernel lengthscale parameter, γ , and the relaxation parameter, C , for the SVM where optimised using a held out validation data set and a grid search with values $C \in \{2^{-5}, 2^{-3}, \dots, 2^{15}\}$ and $\gamma \in \{2^{-15}, 2^{13}, 2^3\}$. Combining both kernel representations using a weighted sum would require optimising 4 parameters simultaneously with cross validation which is not straight-forward using grid search therefore we do not compare against this option.

2.5.5. Results

Table 2.2 compares the misclassification rate of the three methods. Results are averages of a 5-fold cross validation. We can see that the DTW representation is significantly better for characterising the species variations

achieving a better classification accuracy. However, results can be improved by also considering information from the call shape parameters. Moreover, the optimised weights for the kernel combination significantly favour the DTW covariance function with a weight of ≈ 0.8 in contrast to the call shape parameters with weight ≈ 0.2 . If we fix the weight parameters to equal values we obtain a classification error rate of 0.22 ± 0.031 highlighting the importance of the DTW kernel matrix.

The independent length scales allow us also to interpret the discriminatory power of the call shape parameters. In our experiments, the frequency at the center of the duration of a call, the characteristic call frequency (Determined by finding the point in the final 40% of the call having the lowest slope or exhibiting the end of the main trend of the body of the call), as well as the start and end frequencies of the call have consistently obtained a small lengthscale parameter value indicating their importance in species discrimination. This coincides with expert knowledge on bat call shapes where these call shape parameters are extensively used for identifying species.

Table 2.2. Classification results.

Method	Error rate	Std.
SVM shape parameters	0.26	± 0.064
SVM DTW	0.25	± 0.035
GP shape parameters	0.24	± 0.052
GP DTW	0.21	± 0.026
GP DTW + shape param.	0.20	± 0.037

In Figure 2.5 the confusion matrix from the best classification results, 15% misclassification rate, are shown. There is an overall high accuracy for all classes with the exception of species *Lasiurus xanthinus*, class 18, which is often misclassified as *Antrozous pallidus*, class 13, which needs to be further investigated. In contrast, the very similar call shapes of the *Myotis* species are easily discriminated. Finally, misclassification rates are higher to within family species compared to species from other families indicating a common evolutionary path of bat echolocation.

		Confusion Matrix																				
Output Class	Target Class	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
		21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	4.5%	0.0%	4.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
2	0.0%	0.0%	21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100%
3	0.0%	0.0%	4.5%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	80.8%
4	0.0%	0.0%	0.0%	36	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	97.3%
5	0.0%	0.0%	0.0%	0.0%	33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100%
6	0.0%	0.0%	0.0%	0.0%	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	95.2%
7	0.0%	0.0%	0.0%	0.0%	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100%
8	0.0%	0.0%	0.0%	0.0%	21	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	77.8%
9	0.0%	0.0%	0.0%	0.0%	1	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	80.0%
10	0.0%	0.0%	0.0%	0.0%	0	16	0	0	0	0	0	0	0	0	0	3	0	0	0	0	1	80.0%
11	0.0%	0.0%	0.0%	0	5	0	0	9	2	0	0	0	0	0	0	0	0	0	0	0	0	56.2%
12	0.0%	0.0%	0.0%	0	3	1	0	6	22	0	0	0	0	0	0	0	0	0	0	0	0	43.8%
13	0.0%	0.0%	0.0%	0	0	0	0	0	0	13	3	0	0	0	0	2	0	0	0	0	0	72.2%
14	0.0%	0.0%	0.0%	0	0	0	0	0	0	3	19	0	0	0	0	4	0	0	0	0	0	73.1%
15	0.0%	0.0%	0.0%	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	0	0	0	100%
16	0.0%	0.0%	0.0%	0	0	0	0	0	0	0	0	31	0	0	0	6	0	0	0	0	0	83.8%
17	0.0%	0.0%	0.0%	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	100%
18	0.0%	0.0%	0.0%	0	0	0	0	0	0	6	1	0	0	0	7	0	0	0	0	0	0	46.7%
19	0.0%	0.0%	0.0%	0	0	0	0	1	0	0	0	0	0	0	0	0	16	0	0	0	0	94.1%
20	0.0%	0.0%	0.0%	0	2	0	0	0	0	0	0	0	1	0	0	0	15	0	0	0	0	83.3%
21	0.0%	0.0%	0.0%	0	0	0	0	0	0	0	4	0	0	0	1	0	0	0	32	0	0	86.5%
100%	100%	100%	100%	97.1%	100%	100%	67.7%	36.4%	80.0%	60.0%	91.7%	59.1%	67.9%	100%	88.6%	73.3%	50.0%	100%	68.2%	100%	85.0%	
0.0%	0.0%	0.0%	0.0%	2.9%	0.0%	0.0%	32.3%	63.6%	20.0%	40.8%	8.3%	40.9%	32.1%	0.0%	11.4%	26.7%	50.0%	0.0%	31.8%	0.0%	15.0%	

Fig. 2.5. Confusion matrix of the best classification. Classes are in the same order and grouped as in Table 2.1.

A.1. Approximate predictive distribution of latent variables

In the EP approximation the posterior distribution of the latent variables is approximated by a multivariate normal as $\mathcal{N}(\mathbf{f}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with $\boldsymbol{\Sigma} = (\mathbf{K}^{-1} + \tilde{\boldsymbol{\Sigma}}^{-1})^{-1}$ and $\boldsymbol{\mu} = \boldsymbol{\Sigma}\tilde{\boldsymbol{\Sigma}}^{-1}\tilde{\boldsymbol{\mu}}$. The conditional distribution of the GP prior for new latent variables \mathbf{f}_* is also normal, $\mathcal{N}(\mathbf{f}_*|\mathbf{C}\mathbf{f}, \boldsymbol{\Sigma}_*)$ with $\mathbf{C} = \mathbf{k}_*^T \mathbf{K}^{-1}$ and $\boldsymbol{\Sigma}_* = k(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}_*^T \mathbf{K}^{-1} \mathbf{k}_*$. Thus the integral in (2.5) can be written as

$$p(\mathbf{f}_*|\mathbf{x}_*, \mathbf{X}, \mathbf{y}, \hat{\boldsymbol{\theta}}) = \int \mathcal{N}(\mathbf{f}_*|\mathbf{C}\mathbf{f}, \boldsymbol{\Sigma}_*) \mathcal{N}(\mathbf{f}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) d\mathbf{f}.$$

Using the definition of the multivariate normal, expanding and separating terms depending on \mathbf{f} we can get

$$Z_* \int \exp\left(-\frac{1}{2}\mathbf{f}^T(\Sigma^{-1} + \mathbf{C}^T\Sigma_*^{-1}\mathbf{C})\mathbf{f} + (\mathbf{f}_*^T\Sigma_*^{-1}\mathbf{C} + \boldsymbol{\mu}^T\Sigma^{-1})\mathbf{f}\right) d\mathbf{f}$$

where we set

$$Z_* = \frac{\exp\left(-\frac{1}{2}\mathbf{f}_*^T\Sigma_*^{-1}\mathbf{f}_* - \frac{1}{2}\boldsymbol{\mu}^T\Sigma^{-1}\boldsymbol{\mu}\right)}{\sqrt{|2\pi\Sigma_*||2\pi\Sigma|}}$$

The integral is a Gaussian integral and has an analytic solution [16, Chap. 8]

$$Z \exp\left(\frac{1}{2}(\mathbf{f}_*^T\Sigma_*^{-1}\mathbf{C} + \boldsymbol{\mu}^T\Sigma^{-1})(\Sigma^{-1} + \mathbf{C}^T\Sigma_*^{-1}\mathbf{C})^{-1}(\mathbf{C}^T\Sigma_*^{-1}\mathbf{f}_*^T + \Sigma^{-1}\boldsymbol{\mu})\right) \quad (\text{A.1})$$

whith $Z = \sqrt{|2\pi(\Sigma^{-1} + \mathbf{C}^T\Sigma_*^{-1}\mathbf{C})^{-1}|}$

Taking the first term in the exponent of Z_* and the quadratic on \mathbf{f}_* from the exponent in (A.1) and using the matrix inversion lemma we have

$$\begin{aligned} & -\frac{1}{2}\mathbf{f}_*^T(\Sigma_*^{-1} - \Sigma_*^{-1}\mathbf{C}(\Sigma^{-1} + \mathbf{C}^T\Sigma_*^{-1}\mathbf{C})^{-1}\mathbf{C}^T\Sigma_*^{-1})\mathbf{f}_* \\ &= -\frac{1}{2}\mathbf{f}_*^T(\Sigma_* + \mathbf{C}\Sigma\mathbf{C}^T)^{-1}\mathbf{f}_*. \end{aligned}$$

Similarly taking the second term in the exponent of Z_* and the quadratic on $\boldsymbol{\mu}$ from (A.1) we get

$$-\frac{1}{2}\boldsymbol{\mu}^T(\Sigma^{-1} - \Sigma^{-1}(\Sigma^{-1} + \mathbf{C}^T\Sigma_*^{-1}\mathbf{C})^{-1}\Sigma^{-1})$$

where we can use the Searle identity [16, Chap. 3] $\mathbf{A} - \mathbf{A}(\mathbf{A} + \mathbf{B})^{-1}\mathbf{A} = \mathbf{B} - \mathbf{B}(\mathbf{A} + \mathbf{B})^{-1}\mathbf{B}$ and then the matrix inversion lemma to write

$$\begin{aligned} & -\frac{1}{2}\boldsymbol{\mu}^T\mathbf{C}^T(\Sigma_*^{-1} - \Sigma_*^{-1}\mathbf{C}(\Sigma^{-1} + \mathbf{C}^T\Sigma_*^{-1}\mathbf{C})^{-1}\mathbf{C}^T\Sigma_*^{-1})\mathbf{C}\boldsymbol{\mu} \\ &= -\frac{1}{2}\boldsymbol{\mu}^T\mathbf{C}^T(\Sigma_* + \mathbf{C}\Sigma\mathbf{C}^T)^{-1}\mathbf{C}\boldsymbol{\mu}. \end{aligned}$$

For the remaining term in the exponent of (A.1) we can use the special case of the matrix inversion lemma, [16, Chap. 3] $(\mathbf{A}^{-1} + \mathbf{C}^T\mathbf{B}^{-1}\mathbf{C})^{-1}\mathbf{C}^T\mathbf{B}^{-1} = \mathbf{A}\mathbf{C}^T(\mathbf{C}\mathbf{A}\mathbf{C}^T + \mathbf{B})^{-1}$, to get

$$\boldsymbol{\mu}^T\Sigma^{-1}(\Sigma^{-1} + \mathbf{C}^T\Sigma_*^{-1}\mathbf{C})^{-1}\mathbf{C}^T\Sigma^{-1}\mathbf{f}_* = \boldsymbol{\mu}^T\mathbf{C}^T(\Sigma_* + \mathbf{C}\Sigma\mathbf{C}^T)^{-1}\mathbf{f}_*$$

Applying the same logic for the determinant terms in Z_* , Z and collecting all the resulting terms we get

$$\frac{1}{\sqrt{|2\pi(\Sigma_* + \mathbf{C}\Sigma\mathbf{C}^T)|}} \exp\left(-\frac{1}{2}(\mathbf{f}_* - \mathbf{C}\boldsymbol{\mu})^T(\Sigma_* + \mathbf{C}\Sigma\mathbf{C}^T)^{-1}(\mathbf{f}_* - \mathbf{C}\boldsymbol{\mu})\right)$$

which we can recognise as a multivariate normal distribution for \mathbf{f}_* with mean $\mathbf{C}\boldsymbol{\mu}$ and covariance $\Sigma_* + \mathbf{C}\Sigma\mathbf{C}^T$. Substituting \mathbf{C} , $\boldsymbol{\mu}$, Σ and Σ_* we get the expressions in Equations (2.30–2.31) for the mean and covariance of the predictive distribution for \mathbf{f}_* .

A.2. Log of the approximate marginal likelihood

The aim here is to compute the log of the normalisation constant of the approximate distribution in such a way that we avoid numerical over-(under)flows. The aim is to avoid using the site precisions, $\tilde{\Sigma}_n^{-1}$ or \tilde{t}_n , directly which can be arbitrarily small and thus non-invertible. Here we give the general case where the site precisions and locations are respectively matrices and vectors of dimension C . However, the results hold for the binary model with scalar locations and precisions as well as the approximate marginal of the *tilted* distribution for the multinomial case.

The normalisation term of the EP approximate distribution for a GP classification model will have the following form

$$Z_{EP} = \mathcal{N}(\tilde{\mu} | \mathbf{0}, \mathbf{K} + \tilde{\Sigma}) \prod_{n=1}^N \tilde{Z}_n$$

where $\tilde{\mu}$ will be the vector with all mean parameters (in the multinomial case the site mean vectors are concatenated to obtain $\tilde{\mu}$), $\tilde{\Sigma}$ is a matrix formed by the site covariances (in the binary case its a diagonal matrix while in the multinomial case its a block matrix with diagonal blocks) and \tilde{Z}_n are the normalisation terms of the scaled gaussian approximations. Again \tilde{Z}_n has the following form

$$\tilde{Z}_n = \hat{Z}_n \mathcal{N}(\boldsymbol{\mu}_{-n} | \tilde{\mu}_n, \boldsymbol{\Sigma}_{-n} + \tilde{\Sigma}_n)^{-1}$$

where \hat{Z}_n is the 0^{th} moment of the *tilted* distribution, $\boldsymbol{\mu}_{-n}$, $\boldsymbol{\Sigma}_{-n}$ are the mean and covariance of the *cavity* distribution and $\tilde{\mu}_n$, $\tilde{\Sigma}_n$ are the site mean and covariance parameters. The log of the normalisation term then becomes

$$\begin{aligned} \log Z_{EP} &= -\frac{1}{2} \log |\mathbf{K} + \tilde{\Sigma}| - \frac{1}{2} \tilde{\mu}^T (\mathbf{K} + \tilde{\Sigma})^{-1} \tilde{\mu} + \sum_{n=1}^N \frac{1}{2} \log |\boldsymbol{\Sigma}_{-n} + \tilde{\Sigma}_n| \\ &\quad + \sum_{n=1}^N \frac{1}{2} (\boldsymbol{\mu}_{-n} - \tilde{\mu}_n)^T (\boldsymbol{\Sigma}_{-n} + \tilde{\Sigma}_n)^{-1} (\boldsymbol{\mu}_{-n} - \tilde{\mu}_n) + \sum_{n=1}^N \log \hat{Z}_n \end{aligned} \tag{A.2}$$

First we deal with the log determinant terms, i.e. the first and third terms in the above equation which can be written as

$$\begin{aligned}
& -\frac{1}{2} \log |\tilde{\Sigma}(\tilde{\Sigma}^{-1} \mathbf{K} + \mathbf{I})| + \sum_{n=1}^N \frac{1}{2} \log |\tilde{\Sigma}_n(\tilde{\Sigma}_n^{-1} \Sigma_{-n} + \mathbf{I})| \\
& = -\frac{1}{2} \log |\tilde{\Sigma}| - \frac{1}{2} \log |\tilde{\Sigma}^{-1} \mathbf{K} + \mathbf{I}| + \sum_{n=1}^N \frac{1}{2} \log |\tilde{\Sigma}_n| + \frac{1}{2} \log |\tilde{\Sigma}_n^{-1} \Sigma_{-n} + \mathbf{I}| \\
& = -\frac{1}{2} \log |\tilde{\Sigma}^{-1} \mathbf{K} + \mathbf{I}| + \sum_{n=1}^N \frac{1}{2} \log |(\Sigma_n^{-1} - \Sigma_{-n}^{-1}) \Sigma_{-n} + \mathbf{I}| \\
& = -\frac{1}{2} \log |\tilde{\Sigma}^{-1} \mathbf{K} + \mathbf{I}| + \sum_{n=1}^N \frac{1}{2} \log |\Sigma_{-n}| - \frac{1}{2} \log |\Sigma_n|
\end{aligned} \tag{A.3}$$

where in the second line above we use the fact the $\log |\tilde{\Sigma}| = \sum_{n=1}^N \log |\tilde{\Sigma}_n|$ and in the third line we use the relationship $\tilde{\Sigma}_n^{-1} = \Sigma_n^{-1} - \Sigma_{-n}^{-1}$ obtained from the precision of the cavity distribution.

Using the matrix inversion lemma the second term of Equation (A.2) can be written as

$$\begin{aligned}
& -\frac{1}{2} \tilde{\mu}^T \tilde{\Sigma}^{-1} \tilde{\mu} + \frac{1}{2} \tilde{\mu}^T \tilde{\Sigma}^{-1} (\mathbf{K}^{-1} + \tilde{\Sigma}^{-1}) \tilde{\Sigma}^{-1} \tilde{\mu} \\
& = \frac{1}{2} \tilde{v}^T \mu - \frac{1}{2} \sum_{n=1}^N \tilde{\mu}_n^T \tilde{\Sigma}_n^{-1} \tilde{\mu}_n
\end{aligned} \tag{A.4}$$

where we used the relations $\tilde{v} = \tilde{\Sigma}^{-1} \tilde{\mu}$ and $\mu = (\mathbf{K}^{-1} + \tilde{\Sigma}^{-1}) \tilde{\Sigma}^{-1} \tilde{\mu}$.

Collecting the remaining quadratic term from Equation (A.2) and the last term of (A.4) and omitting the summation over n for clarity we get

$$\begin{aligned}
& -\frac{1}{2} \tilde{\mu}_n^T \tilde{\Sigma}_n^{-1} \tilde{\mu}_n + \frac{1}{2} \mu_{-n}^T (\Sigma_{-n} + \tilde{\Sigma}_n)^{-1} \mu_{-n} - \mu_{-n}^T (\Sigma_{-n} + \tilde{\Sigma}_n)^{-1} \tilde{\mu}_n \\
& + \frac{1}{2} \tilde{\mu}_n^T (\Sigma_{-n} + \tilde{\Sigma}_n)^{-1} \tilde{\mu}_n
\end{aligned} \tag{A.5}$$

Applying the matrix inversion lemma the quadratic terms on $\tilde{\mu}_n$ in (A.5) can be written as

$$\begin{aligned}
& -\frac{1}{2} \tilde{\mu}_n^T \tilde{\Sigma}_n^{-1} (\tilde{\Sigma}_n - \tilde{\Sigma}_n (\Sigma_{-n} + \tilde{\Sigma}_n)^{-1} \tilde{\Sigma}_n) \tilde{\Sigma}_n^{-1} \tilde{\mu}_n \\
& = -\frac{1}{2} \tilde{\mu}_n^T \tilde{\Sigma}_n^{-1} (\Sigma_{-n}^{-1} + \tilde{\Sigma}_n^{-1})^{-1} \tilde{\Sigma}_n^{-1} \tilde{\mu}_n
\end{aligned} \tag{A.6}$$

Similarly Applying the matrix inversion lemma on the remaining terms of (A.5), substituting $\tilde{\Sigma}_n^{-1} = \Sigma_n^{-1} - \Sigma_{-n}^{-1}$, $\tilde{\mu}_n = \tilde{\Sigma}_n (\Sigma_n^{-1} \mu_n - \Sigma_{-n}^{-1} \mu_{-n})$,

obtained from the relations for the covariance and mean of the *cavity*, and canceling terms out (A.5) can be simplified to

$$\frac{1}{2}\boldsymbol{\mu}_{-n}^T \boldsymbol{\Sigma}_{-n}^{-1} \boldsymbol{\mu}_{-n} - \frac{1}{2}\boldsymbol{\mu}_n^T \boldsymbol{\Sigma}_n^{-1} \boldsymbol{\mu}_n \quad (\text{A.7})$$

Collecting all remaining terms from (A.2, A.3, A.4, A.7) the log of the approximate marginal can be written as

$$\begin{aligned} \log Z_{EP} = & \frac{1}{2}\tilde{\mathbf{v}}^T \boldsymbol{\mu} - \frac{1}{2}\log |\tilde{\boldsymbol{\Sigma}}^{-1} \mathbf{K} + \mathbf{I}| + \frac{1}{2} \sum_{n=1}^N (\boldsymbol{\mu}_{-n}^T \boldsymbol{\Sigma}_{-n}^{-1} \boldsymbol{\mu}_{-n} + \log |\boldsymbol{\Sigma}_{-n}|) \\ & - \frac{1}{2} \sum_{n=1}^N (\boldsymbol{\mu}_n^T \boldsymbol{\Sigma}_n^{-1} \boldsymbol{\mu}_n + \log |\boldsymbol{\Sigma}_n|) + \sum_{n=1}^N \log \hat{Z}_n \end{aligned} \quad (\text{A.8})$$

The determinant $|\tilde{\boldsymbol{\Sigma}}^{-1} \mathbf{K} + \mathbf{I}|$ can be computed efficiently by exploiting the structure of $\tilde{\boldsymbol{\Sigma}}^{-1}$. See Section A.4.

A.3. Marginal covariance and mean of the *tilted* distribution

The approximate marginal covariance of the *tilted* distribution for \mathbf{f}_n is $\hat{\boldsymbol{\Sigma}}_n = \mathbf{H}^T (\boldsymbol{\Sigma}_{\mathbf{w}_n}^{-1} + \tilde{\mathbf{B}}_n \tilde{\mathbf{T}}_n \tilde{\mathbf{B}}_n^T)^{-1} \mathbf{H}$. Using the matrix inversion lemma we can write it as

$$\hat{\boldsymbol{\Sigma}}_n = \mathbf{H}^T \boldsymbol{\Sigma}_{\mathbf{w}_n} \mathbf{H} - \mathbf{H}^T \boldsymbol{\Sigma}_{\mathbf{w}_n} \tilde{\mathbf{B}}_n (\tilde{\mathbf{T}}_n^{-1} + \tilde{\mathbf{B}}_n^T \boldsymbol{\Sigma}_{\mathbf{w}_n} \tilde{\mathbf{B}}_n)^{-1} \tilde{\mathbf{B}}_n^T \boldsymbol{\Sigma}_{\mathbf{w}_n} \mathbf{H}$$

where the first term is $\boldsymbol{\Sigma}_{-n}$ by the definition of $\boldsymbol{\Sigma}_{\mathbf{w}_n}$ in Equation (2.22). For the second term we define $\mathbf{B}_n = \mathbf{H}^T \tilde{\mathbf{B}}_n$ and use the fact that by definition the last row of $\tilde{\mathbf{B}}_n$ is a row of ones, the last row of \mathbf{H} is zeros and $\boldsymbol{\Sigma}_{\mathbf{w}_n}$ is a block matrix composed of $\boldsymbol{\Sigma}_{-1}$ and 1 and thus we can multiply with $\mathbf{H} \mathbf{H}^T$ where necessary to extract $\boldsymbol{\Sigma}_{-1}$ and inside the parenthesis add the remaining matrix of ones. Therefore the second term can be written as

$$\boldsymbol{\Sigma}_{-n} \mathbf{B}_n (\tilde{\mathbf{T}}_n^{-1} + \mathbf{1} \mathbf{1}^T + \mathbf{B}_n^T \boldsymbol{\Sigma}_{-n} \mathbf{B}_n)^{-1} \mathbf{B}_n^T \boldsymbol{\Sigma}_{-n}$$

where $\mathbf{1}$ is a $(C - 1) \times 1$ vector of ones. Collecting both terms and using the matrix inversion lemma we can write $\hat{\boldsymbol{\Sigma}}_n$ as

$$\hat{\boldsymbol{\Sigma}}_n = (\boldsymbol{\Sigma}_{-n}^{-1} + \mathbf{B}_n (\tilde{\mathbf{T}}_n^{-1} + \mathbf{1} \mathbf{1}^T)^{-1} \mathbf{B}_n^T)^{-1}.$$

Since the update for the approximate precision parameters in EP is $\tilde{\boldsymbol{\Sigma}}_n^{-1} = \hat{\boldsymbol{\Sigma}}_n^{-1} - \boldsymbol{\Sigma}_{-n}^{-1}$, using the above result it can be written as

$$\tilde{\boldsymbol{\Sigma}}_n^{-1} = \mathbf{B}_n (\tilde{\mathbf{T}}_n^{-1} + \mathbf{1} \mathbf{1}^T)^{-1} \mathbf{B}_n^T, \quad (\text{A.9})$$

Equation (A.9) can be further simplified [5] by writing $\mathbf{B}_n = -\mathbf{A}_n \mathbf{E}_{-y_n}$ with $\mathbf{A}_n = [\mathbf{I}_C - \mathbf{e}_{y_n} \mathbf{1}_C^T]$ to get

$$\tilde{\Sigma}_n^{-1} = \mathbf{A}_n (\mathbf{E}_{-y_n} \tilde{\mathbf{T}}_n \mathbf{E}_{-y_n}^T - \boldsymbol{\pi}_n (\mathbf{1}_C^T \boldsymbol{\pi}_n)^{-1} \boldsymbol{\pi}_n^T) \mathbf{A}_n^T$$

where $\boldsymbol{\pi}_n = \mathbf{E}_{-y_n} \tilde{\mathbf{a}}_n - \mathbf{e}_{y_n}$ and used $\mathbf{B}_n \tilde{\mathbf{a}}_n = -\mathbf{A}_n \boldsymbol{\pi}_n$. We can use the fact that $\mathbf{A}_n \mathbf{e}_{y_n} = 0$ and add $\mathbf{e}_{y_n} \mathbf{e}_{y_n}^T$ inside the brackets to the first term to obtain

$$\tilde{\Sigma}_n^{-1} = \text{diag}(\boldsymbol{\pi}_n) - (\mathbf{1}_C^T \boldsymbol{\pi}_n)^{-1} \boldsymbol{\pi}_n \boldsymbol{\pi}_n^T.$$

In a similar way we can obtain the approximate marginal mean of the *tilted* distribution for the latent variables \mathbf{f} as

$$\hat{\mu}_n = \mathbf{H}^T (\Sigma_{w_n}^{-1} + \tilde{\mathbf{B}}_n \tilde{\mathbf{T}}_n \tilde{\mathbf{B}}_n^T)^{-1} (\Sigma_{w_n}^{-1} \boldsymbol{\mu}_{w_n} + \tilde{\mathbf{B}}_n \tilde{\boldsymbol{\beta}}_n).$$

In the first term of the r.h.s. we can again multiply with $\mathbf{H} \mathbf{H}^T$ where necessary to extract the components Σ_{-n} and $\boldsymbol{\mu}_{-n}$ from Σ_{w_n} and $\boldsymbol{\mu}_{w_n}$ without changing the result. The first term then becomes $\hat{\Sigma}_n \Sigma_{-n}^{-1} \boldsymbol{\mu}_{-n}$. In the second term we can pre-multiply $\tilde{\boldsymbol{\beta}}_n$ with $\tilde{\mathbf{T}}_n \tilde{\mathbf{T}}_n^{-1}$ and then use the matrix inversion lemma to write

$$\begin{aligned} \mathbf{H}^T (\Sigma_{w_n}^{-1} + \tilde{\mathbf{B}}_n \tilde{\mathbf{T}}_n \tilde{\mathbf{B}}_n^T)^{-1} \tilde{\mathbf{B}}_n \tilde{\mathbf{T}}_n \tilde{\mathbf{T}}_n^{-1} \tilde{\boldsymbol{\beta}}_n = \\ \mathbf{H}^T \Sigma_{w_n} \tilde{\mathbf{B}}_n (\tilde{\mathbf{T}}_n^{-1} + \tilde{\mathbf{B}}_n^T \Sigma_{w_n} \tilde{\mathbf{B}}_n)^{-1} \tilde{\mathbf{T}}_n^{-1} \tilde{\boldsymbol{\beta}}_n \end{aligned}$$

where we can again multiply with $\mathbf{H} \mathbf{H}^T$ as we did for the covariance matrix to get

$$\Sigma_{-n} \mathbf{B}_n (\tilde{\mathbf{T}}_n^{-1} + \mathbf{1} \mathbf{1}^T + \mathbf{B}_n^T \Sigma_{-n} \mathbf{B}_n)^{-1} \tilde{\mathbf{T}}_n^{-1} \tilde{\boldsymbol{\beta}}_n.$$

Collecting the terms we can get the *tilted* mean for the latent variables to be

$$\hat{\mu}_n = \hat{\Sigma}_n \Sigma_{-n}^{-1} \boldsymbol{\mu}_{-n} + \Sigma_{-n} \mathbf{B}_n (\tilde{\mathbf{T}}_n^{-1} + \mathbf{1} \mathbf{1}^T + \mathbf{B}_n^T \Sigma_{-n} \mathbf{B}_n)^{-1} \tilde{\mathbf{T}}_n^{-1} \tilde{\boldsymbol{\beta}}_n.$$

The EP update for the approximate mean parameters is $\tilde{\boldsymbol{\mu}}_n = \tilde{\Sigma}_n (\hat{\Sigma}_n^{-1} \hat{\boldsymbol{\mu}}_n - \Sigma_{-n} \boldsymbol{\mu}_{-n})$. Substituting $\hat{\boldsymbol{\mu}}_n$ with the above expression and cancelling terms out we get

$$\tilde{\boldsymbol{\mu}}_n = \tilde{\Sigma}_n \hat{\Sigma}_n^{-1} \Sigma_{-n} \mathbf{B}_n (\tilde{\mathbf{T}}_n^{-1} + \mathbf{1} \mathbf{1}^T + \mathbf{B}_n^T \Sigma_{-n} \mathbf{B}_n)^{-1} \tilde{\mathbf{T}}_n^{-1} \tilde{\boldsymbol{\beta}}_n.$$

Using the matrix inversion lemma for the inverse matrix in the parenthesis, substituting $\hat{\Sigma}_n^{-1}$ and canceling terms out we get

$$\tilde{\boldsymbol{\mu}}_n = \tilde{\Sigma}_n \mathbf{B}_n (\tilde{\mathbf{T}}_n^{-1} + \mathbf{1} \mathbf{1}^T)^{-1} \tilde{\mathbf{T}}_n^{-1} \tilde{\boldsymbol{\beta}}_n. \quad (\text{A.10})$$

As we did with Equation (A.9) we can simplify further to get the location parameters

$$\tilde{\boldsymbol{v}}_n = (\mathbf{1}^T \tilde{\boldsymbol{\beta}}_n) (\mathbf{1}^T \boldsymbol{\pi}_n)^{-1} \boldsymbol{\pi}_n - \mathbf{E}_{-y_n} \tilde{\boldsymbol{\beta}}_n.$$

A.4. Covariance of the approximate distribution for the multinomial probit EP

Using the matrix inversion lemma the covariance of the approximate distribution can be written as

$$\boldsymbol{\Sigma} = \mathbf{K} - \mathbf{K}\mathbf{M}\mathbf{K}^T,$$

where $\mathbf{M} = (\mathbf{K} + \tilde{\boldsymbol{\Sigma}})^{-1}$. We can use the matrix inversion lemma to invert the matrix of site precisions

$$\begin{aligned}\tilde{\boldsymbol{\Sigma}} &= (\mathbf{D} - \mathbf{D}\mathbf{R}(\mathbf{R}^T\mathbf{D}\mathbf{R})^{-1}\mathbf{R}^T\mathbf{D})^{-1} \\ &= \mathbf{D}^{-1} + \mathbf{D}^{-1}\mathbf{D}\mathbf{R}(\mathbf{R}^T\mathbf{D}\mathbf{R} - \mathbf{R}^T\mathbf{D}\mathbf{D}^{-1}\mathbf{D}\mathbf{R})^{-1}\mathbf{R}^T\mathbf{D}\mathbf{D}^{-1} \\ &= \mathbf{D}^{-1} + \mathbf{R}\mathbf{0}^{-1}\mathbf{R}^T\end{aligned}$$

Substituting the expression for $\tilde{\boldsymbol{\Sigma}}$ back into \mathbf{M} and using the matrix inversion lemma once more we get

$$\begin{aligned}\mathbf{M} &= (\mathbf{K} + \mathbf{D}^{-1} + \mathbf{R}\mathbf{0}^{-1}\mathbf{R}^T)^{-1} \\ &= \mathbf{B} - \mathbf{B}\mathbf{R}(\mathbf{R}^T\mathbf{B}\mathbf{R})^{-1}\mathbf{R}^T\mathbf{B},\end{aligned}$$

where \mathbf{B} is

$$\begin{aligned}\mathbf{B} &= (\mathbf{K} + \mathbf{D}^{-1})^{-1} = (\mathbf{D}^{-1}(\mathbf{K}\mathbf{D} + \mathbf{I}))^{-1} \\ &= \mathbf{D}^{1/2}(\mathbf{D}^{1/2}\mathbf{K}\mathbf{D}^{1/2} + \mathbf{I})^{-1}\mathbf{D}^{1/2},\end{aligned}$$

where we use the fact that \mathbf{D} is a diagonal matrix.

Similarly the determinant of the covariance matrix as well as the determinant term of the log marginal in Equation (A.8) can be computed efficiently using

$$|\mathbf{K}^{-1} + \tilde{\boldsymbol{\Sigma}}^{-1}| = |\mathbf{K}|^{-1}|\mathbf{I} + \mathbf{K}\tilde{\boldsymbol{\Sigma}}^{-1}| = |\mathbf{K}|^{-1}|\mathbf{A}||\mathbf{R}^T\mathbf{D}\mathbf{R}^T|^{-1}|\mathbf{R}^T\mathbf{B}\mathbf{R}|$$

where $\mathbf{A} = \mathbf{I} + \mathbf{D}^{1/2}\mathbf{K}\mathbf{D}^{1/2}$.

References

- [1] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press (2005). ISBN 026218253X.
- [2] C. Leslie, R. Kuang, and K. Bennett, Fast string kernels using inexact matching for protein sequences, *Journal of Machine Learning Research*. 5, 1435–1455 (2004).

- [3] M. Girolami and S. Rogers, Variational bayesian multinomial probit regression with gaussian process priors, *Neural Computation*. **18**(8), 1790–1817 (2006).
- [4] M. Kuss and C. E. Rasmussen, Assessing approximate inference for binary gaussian process classification., *Journal of Machine Learning Research*. pp. 1679–1704 (2005).
- [5] J. Riihimaki, P. Jylanki, and A. Vehtari, Nested expectation propagation for gaussian process classification with a multinomial probit likelihood, *Journal of Machine Learning Research*. **14**, 75–109 (2013).
- [6] T. Minka, Expectation propagation for approximate bayesian inference. In *Proceedings of the Seventeenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pp. 362–369, Morgan Kaufmann, San Francisco, CA (2001).
- [7] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2006). ISBN 0387310738.
- [8] M. Girolami and M. Zhong, Data integration for classification problems employing gaussian process priors. In eds. B. Schölkopf, J. Platt, and T. Hoffman, *Advances in Neural Information Processing Systems 19*, pp. 465–472. MIT Press, Cambridge, MA (2007).
- [9] M. Seeger and M. I. Jordan, Sparse gaussian process classification with multiple classes. Technical report, University of California, Berkeley (2004).
- [10] H.-C. Kim and Z. Ghahramani, Bayesian gaussian process classification with the EM-EP algorithm, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **28**(12), 1948–1959 (2006). ISSN 0162-8828. doi: <http://doi.ieeecomputersociety.org/10.1109/TPAMI.2006.238>.
- [11] V. Stathopoulos, V. Zamora-Gutierrez, K. E. Jones, and M. Girolami, Bat call identification with gaussian process multinomial probit regression and a dynamic time warping kernel. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014*, pp. 913–921 (2014). URL <http://jmlr.org/proceedings/papers/v33/stathopoulos14.html>.
- [12] C. L. Walters, R. Freeman, A. Collen, c. Dietz, M. Brock Fenton, G. Jones, M. K. Obrist, S. J. Puechmaille, T. Sattler, B. M. Siemers, S. Parsons, and K. E. Jones, A continental-scale tool for acoustic identification of European bats, *Journal of Applied Ecology*. **49**(5), 1064–1074 (2012). ISSN 00218901.
- [13] H. Sakoe and S. Chiba, Dynamic programming algorithm optimization for spoken word recognition, *IEE Transactions on Acoustics, Speech and Signal Processing*. **26**, 43–49 (1978).
- [14] T. Damoulas, S. Henry, A. Farnsworth, M. Lanzone, and C. Gomes, Bayesian classification of flight calls with a novel dynamic time warping kernel. In *Proceedings of the 2010 Ninth International Conference on Machine Learning and Applications, ICMLA '10*, pp. 424–429, IEEE Computer Society, Washington, DC, USA (2010). ISBN 978-0-7695-4300-0. doi: [10.1109/ICMLA.2010.69](https://doi.org/10.1109/ICMLA.2010.69).
- [15] C.-C. Chang and C.-J. Lin, LIBSVM: A library for support vector ma-

- chines, *ACM Transactions on Intelligent Systems and Technology*. **2**, 27:1–27:27 (2011). Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [16] K. B. Petersen and M. S. Pedersen. The matrix cookbook. URL <http://www2.imm.dtu.dk/pubdb/p.php?3274> (nov, 2012). Version 20121115.

Chapter 3

Active Multitask Learning using Supervised and Shared Latent Topics

Ayan Acharya¹, Raymond J. Mooney² and Joydeep Ghosh¹

¹*Department of Electrical and Computer Engineering*

University of Texas, Austin, USA

aacharya@utexas.edu

²*Department of Computer Science*

University of Texas, Austin, USA

Multitask learning (MTL) is a machine learning technique where a problem is learnt together with other related problems at the same time, using a shared representation, so that information can be exchanged across related problems. MTL has been adopted quite frequently to alleviate problems with sparsity of labeled data across different learning tasks. Active learning, on the other hand, reduces the cost of labeling examples by making informative queries over an unlabeled pool of data. Therefore, a unification of both of these approaches can potentially be useful in settings where labeled information is expensive to obtain but the learning tasks or domains have some common characteristics. This chapter introduces frameworks that integrate MTL and active learning which make use of both latent and supervised shared topics to accomplish MTL. Experimental results on both document and image classification show that integrating MTL and active learning along with shared latent and supervised topics is superior to other methods that do not employ all of these components.

3.1. Introduction

Humans can distinguish as many as 30,000 relevant object classes [1]. Training an isolated object detector for each of these different classes would require millions of training examples in aggregate. Computer vision researchers have proposed a more efficient learning mechanism in which object categories are learned via *shared* attributes, abstract descriptors of object properties such as “striped” or “has four legs” [2–4]. The attributes serve as an intermediate layer in a classifier cascade. The classifier in the first stage is trained to predict the attributes from the

raw features and that in the second stage is trained to predict the categories from the attributes. During testing, only the raw features are observed and the attributes must be inferred. This approach is inspired by human perception and learning from high-level object descriptions. For example, from the phrase “eight-sided red traffic sign with white writing”, humans can detect stop signs [3]. Similarly, from the description “large gray animals with long trunks”, human can identify elephants. If the *shared* attributes transcend object class boundaries, such a classifier cascade is beneficial for *transfer learning* [5] (see Section 3.2.2) where fewer labeled examples are available for some object categories compared to others [3]. This representation is illustrated in Fig. 3.1.

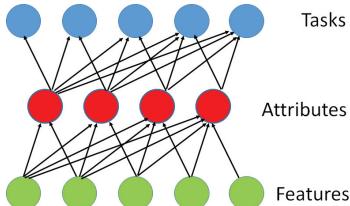


Fig. 3.1. MTL with Shared Attributes.

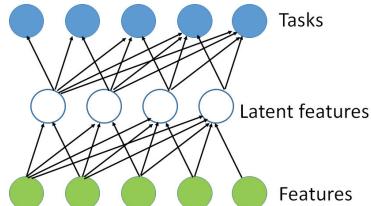


Fig. 3.2. MTL with Multi-layer Perceptron.

Multitask learning (MTL) is a form of transfer learning in which simultaneously learning multiple related “tasks” allows each one to benefit from the learning of all of the others. If the tasks are related, training one task should provide helpful “inductive bias” for learning the other tasks. To enable the reuse of training information across multiple related tasks, all tasks might utilize the same latent shared intermediate representation – for example, a shared hidden layer in a multi-layer perceptron [6] (as shown in Fig. 3.2). In this case, the training examples for all tasks provide good estimates of the weights connecting the input layer to the hidden layer, and hence only a small number of examples per task is sufficient to achieve high accuracy. This approach is in contrast to “isolated” training of tasks where each task is learned independently using a separate classifier.

In this chapter, our objective is to combine these two approaches to build an MTL framework that can use *both* attributes *and* class labels. To alleviate problems related to sparsity of labeled information, we further combine MTL with active learning techniques to efficiently query over both attributes and class labels. The multiple tasks in such setting correspond to different object categories (classes), and *both* observable attributes and latent properties are shared across the tasks. We want to emphasize that the proposed frameworks support general MTL; however, the datasets we use happen to be multiclass, where each class is treated

as a separate “task” (as typical in multi-class learning based on binary classifiers). But, in no way are the frameworks restricted to multiclass MTL. Since attribute-based learning has been shown to support effective transfer learning in computer vision, the tasks here naturally correspond to object classes.

The rest of the chapter is organized as follows. We present related literature in Section 3.2, followed by the descriptions of Latent Dirichlet allocation (LDA) models DSLDA and NP-DSLDA, two MTL frameworks in Section 3.3. In Section 3.4 these two frameworks are combined with active learning techniques. Experimental results on both multi-class image and document categorization are presented in Section 3.5, demonstrating the value of integrating MTL, active learning, attributes and latent shared representations. Finally, future directions and conclusions are presented in Section 3.6.

Note on Notation: Vectors and matrices are denoted by bold-faced lowercase and capital letters, respectively. Scalar variables are written in italic font, and sets are denoted by calligraphic uppercase letters. $\text{Dir}()$, $\text{Beta}()$ and $\text{multinomial}()$ stand for Dirichlet, Beta and multinomial distribution respectively.

3.2. Background

In this section, a brief introduction to probabilistic topic models, transfer learning and active learning is provided. We also present a brief description of incremental EM algorithm (in Section 3.2.5) and online support vector machine (in Section 3.2.6) which are useful for incremental learning during active selection of labels.

3.2.1. Probabilistic topic models

Latent Dirichlet allocation (LDA) [7] treats documents as a mixture of topics, which in turn are defined by a distribution over a set of words. The words in a document are assumed to be sampled from multiple topics. In its original formulation, LDA can be viewed as a purely-unsupervised form of dimensionality reduction and clustering of documents in the topic space (see Fig. 3.4). The graphical model of LDA is shown in Fig. 3.3. The generative process of LDA is described below:

- For the n^{th} document, sample a topic selection probability vector $\boldsymbol{\theta}_n \sim \text{Dir}(\boldsymbol{\alpha})$, where $\boldsymbol{\alpha}$ is the parameter of a Dirichlet distribution of dimension K , which is the total number of topics.
- For the m^{th} word in the n^{th} document, sample a topic $z_{nm} \sim \text{multinomial}(\boldsymbol{\theta}_n)$.
- Sample the word $w_{nm} \sim \text{multinomial}(\boldsymbol{\beta}_{z_{nm}})$, where $\boldsymbol{\beta}_k$ is a multinomial distribution over the vocabulary of words corresponding to the k^{th} topic.

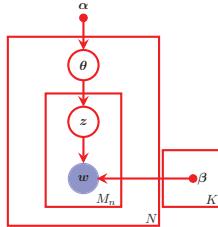


Fig. 3.3. Graphical Model of LDA.

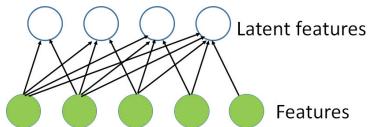


Fig. 3.4. Illustration of LDA.

Several extensions of LDA have subsequently incorporated some sort of supervision. Some approaches provide supervision by labeling each document with its set of topics [8, 9]. In particular, in *Labeled LDA* (LLDA) [8], the primary objective is to build a model of the words that indicate the presence of certain topic labels (see Fig. 3.6). For example, when a user explores a webpage based on certain tags, LLDA can be used to highlight interesting portions of the page or build a summary of the text from multiple webpages that share the same set of tags. The words in a given training document are assumed to be sampled *only* from the supervised topics, which the document has been labeled as covering. The graphical model of LLDA is shown in Fig. 3.5 where Λ is a binary vector representing the presence or absence of topics in a training document. For test data, however, topics are not observed and one (ideally) needs to explore all possible configurations of topics and infer the best one.

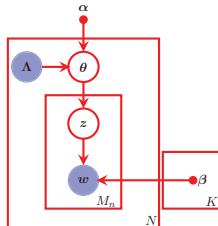


Fig. 3.5. Graphical Model of LLDA.

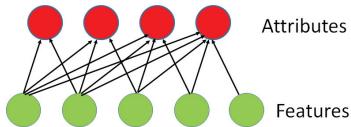


Fig. 3.6. Illustration of LLDA.

Some other researchers [10–12] assume that supervision is provided for a single *response variable* to be predicted for a given document. The response variable might be real-valued or categorical, and modeled by a normal, Poisson, Bernoulli, multinomial or other distribution (see [12] for details). Some examples of documents with response variables are essays with their grades, movie reviews with their numerical ratings, web pages with their number of hits over a certain period

of time, and documents with category labels. In *Maximum Entropy Discriminative LDA* (MedLDA) [11], the objective is to infer some low-dimensional (topic-based) representation of documents which is predictive of the response variable. Essentially, MedLDA solves two problems jointly – dimensionality reduction and max-margin classification using the features in the dimensionally-reduced space (see Fig. 3.8). Compared to earlier versions of supervised topic models [10, 12], MedLDA has simpler update equations and produces superior experimental results. Therefore, in the frameworks presented in Sections 3.3.2 and 3.3.4, the max-margin principle adopted in MedLDA is preferred over other supervised topic models.

In MedLDA, the topic space representation of the documents is treated as features for an SVM learning framework. In particular, for the n^{th} document, we generate $Y_n = \arg \max_y \mathbf{r}_y^T \mathbb{E}(\bar{\mathbf{z}}_n)$ where Y_n is the class label associated with the n^{th} document, $\bar{\mathbf{z}}_n = \sum_{m=1}^{M_n} \mathbf{z}_{nm} / M_n$. Here, \mathbf{z}_{nm} is an indicator vector of dimension K . \mathbf{r}_y is a K -dimensional real vector corresponding to the y^{th} class, and it is assumed to have a prior distribution $\mathcal{N}(0, 1/C)$. M_n is the number of words in the n^{th} document. The maximization problem to generate Y_n (or the classification problem) is carried out using a max-margin principle – the exact formulation of which will be discussed later using variational approximation. Since MedLDA includes discriminative modeling of the class labels, it is not possible to draw a plate model. However, for the ease of understanding, in Fig. 3.7, we show a representative plate model with the discriminative part denoted by dotted lines.

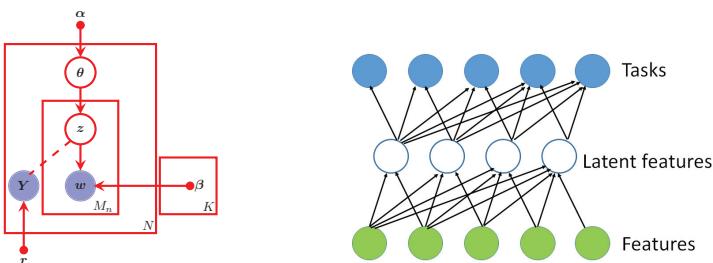
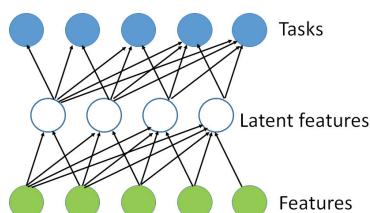


Fig. 3.7. Graphical Model of MedLDA.

Fig. 3.8. Illustration of MedLDA.



3.2.2. Transfer and multitask learning

Transfer learning allows the learning of some tasks to benefit the learning of others through either simultaneous [6] or sequential [13] training. In multitask learning (MTL [6]), a single model is simultaneously trained to perform multiple related tasks. MTL has emerged as a very promising research direction for various applications including biomedical informatics [14], marketing [15], natural language processing [16], and computer vision [17].

Many different MTL approaches have been proposed over the past 15 years (*e.g.*, see [5, 18, 19] and references therein). These include different learning methods, such as empirical risk minimization using group-sparse regularizers [20–22], hierarchical Bayesian models [23, 24] and hidden conditional random fields [25]. Evgeniou *et al.* [26] proposed the regularized MTL which constrained the models of all tasks to be close to each other. The task relatedness in MTL has also been modeled by constraining multiple tasks to share a common underlying structure [6, 27, 28]. Ando and Zhang [29] proposed a structural learning formulation, which assumed multiple predictors for different tasks shared a common structure on the underlying predictor space.

In all of the MTL formulations mentioned above, the basic assumption is that all tasks are related. In practical applications, these might not be the case and the tasks might exhibit a more sophisticated group structure. Such structure is handled using clustered multi-task learning (CMTL). In [30] CMTL is implemented by considering a mixture of Gaussians instead of single Gaussian priors. Xue *et al.* [31] introduced the Dirichlet process prior that automatically identifies sub-groups of related tasks. In [32], a clustered MTL framework was proposed that simultaneously identified clusters and performed multi-task inference.

In the models presented in this chapter, an LDA-based approach to MTL is easily obtained by maintaining a common set of topics to support the prediction of multiple response variables. This idea is analogous to implementing MTL using a common shared underlying structure [6, 27, 28].

3.2.3. Active learning via Expected Error Reduction

Of the several measures for selecting labels in active learning algorithms, a decision-theoretic approach called Expected Error Reduction [33] has been used quite extensively in practice [4, 34]. This approach aims to measure how much the generalization error of a model is likely to be reduced based on some labeled information y of an instance x taken from the unlabeled pool \mathcal{U} . The idea is to estimate the expected future error of a model trained using $\mathcal{L} \cup \langle x, y \rangle$ on the re-

maining unlabeled instances in \mathcal{U} , and query the instance with minimal expected future error. Here \mathcal{L} denotes the labeled pool of data. One approach is to minimize the expected 0/1 loss:

$$\mathbf{x}_{0/1}^* = \operatorname{argmax}_{\mathbf{x}} \sum_n P_{\kappa}(y_n | \mathbf{x}) \left(\sum_{u=1}^U 1 - P_{\kappa^{+(\mathbf{x}, y_n)}}(\hat{y}, \mathbf{x}^{(u)}) \right), \quad (3.1)$$

where $\kappa^{+(\mathbf{x}, y_n)}$ refers to the new model after it has been re-trained with the training set $\mathcal{L} \cup \langle \mathbf{x}, y_n \rangle$. Note that we do not know the true label for each query instance, so we approximate using expectation over all possible labels under the current model. The objective is to reduce the expected number of incorrect predictions.

3.2.4. Active knowledge transfer

There has been some effort to integrate active and transfer learning in the same framework. In [35] the authors utilized a maximum likelihood classifier to learn parameters from the source domain and use these parameters to seed the EM algorithm that explains the unlabeled data in the target domain. The example which contributed to maximum expected KL divergence of the posterior distribution with the prior distribution was selected in the active step. In [36], the source data is first used to train a classifier, the parameters of which are later updated in an online manner with new examples actively selected from the target domain. The active selection criterion is based on uncertainty sampling [34]. Similarly, in [37], a naïve Bayes classifier is first trained with examples from the source domain and then incrementally updated with data from the target domain selected using uncertainty sampling. The method proposed in [38] maintains a classifier trained on the source domain(s) and the prediction of this classifier is trusted only when the likelihood of the data in the target domain is sufficiently high. In case of lower likelihood, domain experts are asked to label the example. Harpale & Young [39] proposed active multitask learning for adaptive filtering [40] where the underlying classifier is logistic regression with Dirichlet process priors. Any feedback provided in the active selection phase improves both the task-specific and the global performance *via* a measure called *utility gain* [39]. Saha *et al.* [41] formulated an online active multitask learning framework where the information provided for one task is utilized for other tasks through a task correlation matrix. The updates are similar to perceptron updates. For active selection, they use a margin based sampling scheme which is a modified version of the sampling scheme used in [42].

In contrast to this previous work, our approach employs a topic-modeling framework and uses expected error reduction for active selection. Such an ac-

tive selection mechanism necessitates fast incremental update of model parameters, and hence the inference and estimation problems become challenging. This approach to active selection is more immune to noisy observations compared to simpler methods such as uncertainty sampling [34]. Additionally, our approach can query both class labels *and* supervised topics (i.e. attributes), which has not previously been explored in the context of MTL.

3.2.5. Incremental EM algorithm

The EM algorithm proposed by Dempster *et al.* [43] can be viewed as a joint maximization problem over $q(\cdot)$, the conditional distribution of the hidden variables \mathbf{Z} given the model parameters κ and the observed variables \mathbf{X} . The relevant objective function is given as follows:

$$F(q, \kappa) = \mathbb{E}_q[\log(p(\mathbf{X}, \mathbf{Z}|\kappa))] + H(q), \quad (3.2)$$

where $H(q)$ is the entropy of the distribution $q(\cdot)$. Often, $q(\cdot)$ is restricted to a family of distributions \mathcal{Q} . It can be shown that if θ^* is the maximizer of the above objective F then it also maximizes the likelihood of the observed data. In most of the models used in practice, the joint distribution is assumed to factorize over the N instances implying that $p(\mathbf{X}, \mathbf{Z}|\kappa) = \prod_{n=1}^N p(\mathbf{x}_n, \mathbf{z}_n|\kappa)$. One can further restrict the family of distributions \mathcal{Q} to maximize over in Eq. (3.2) to the factorized form:

$$q(\mathbf{Z}) = \prod_{n=1}^N q(\mathbf{z}_n|\mathbf{x}_n) = \prod_{n=1}^N q_n. \quad (3.3)$$

An incremental variant of the EM algorithm that exploits such separability structure in both $p(\cdot)$ and $q(\cdot)$ was first proposed by Neal & Hinton [44]. Under such structure, the objective function in Eq. (3.2) decomposes over the observations $F(q, \theta) = \sum_{n=1}^N F_n(q_n, \kappa)$, and the following incremental algorithm can instead be used to maximize F :

- **E step:** Choose some observation n to be updated over, set $q_{n'}^{(t)} = q_{n'}^{(t-1)}$ for $n' \neq n$ (no update) and set $q_n^{(t)} = \operatorname{argmax}_{q_n} F_n(q_n, \kappa^{(t-1)})$.
- **M step:** $\kappa^{(t)} = \operatorname{argmax}_{\kappa} F(q^{(t)}, \kappa)$.

3.2.6. Online Support Vector Machines (SVMs)

The online SVM proposed by Bordes *et al.* [45, 46] has three distinct modules that work in unison to provide a scalable learning mechanism. These modules are

named “ProcessNew”, “ProcessOld” and “Optimize”. All of these modules use a common operation called “SMOStep” and the memory footprint is limited to the support vectors and associated gradient information. The module “ProcessNew” operates on a pattern that is not a support pattern. In such an update, one of the classes is chosen as the label of the support pattern and the other class is chosen such that it defines a feasible direction with the highest gradient. It then performs an SMO step with the example and the selected classes. The module “ProcessOld” randomly picks a support pattern and chooses two classes that define the feasible direction with the highest gradient for that support pattern. “Optimize” resembles “ProcessOld” but picks two classes among those that correspond to existing support vectors.

3.3. Multitask Learning using Both Supervised and Shared Latent Topics

3.3.1. Task definition

Assume we are given a training corpus consisting of N documents belonging to Y different classes (where each document belongs to exactly one class and each class corresponds to a different task). Further assume that each of these training documents is also annotated with a set of K_2 different topic “tags” (henceforth referred to as “supervised topics”). For computer vision data, the supervised topics correspond to the attributes provided by human experts. The objective is to train a model using the words in a data, as well as the associated supervised topic tags and class labels, and then use this model to classify completely unlabeled test data for which no topic tags nor class labels are provided. The human-provided supervised topics are presumed to provide abstract information that is helpful in predicting the class labels of test documents.

3.3.2. Doubly supervised LDA (DSLDA)

In order to include both types of supervision (class and topic labels), a combination of the approaches described in Section 3.2.1 is proposed. Note that LLDA uses *only* supervised topics and does not have any mechanism for generating class labels. On the other hand, MedLDA has only *latent* topics but learns a discriminative model for predicting classes from these topics. To the best of our knowledge, ours is the first LDA approach to integrate both types of supervision in a single framework. The generative process of DSLDA is described below.

For the n^{th} document, sample a topic selection probability vector $\theta_n \sim \text{Dir}(\alpha_n)$, where $\alpha_n = \Lambda_n \alpha$ and α is the parameter of a Dirichlet distribution

of dimension K , which is the total number of topics. The topics are assumed to be of two types – latent and supervised, and there are K_1 latent topics and K_2 supervised topics. Therefore, $K = K_1 + K_2$. Latent topics are never observed, while supervised topics are observed in training but not in test data. Henceforth, in each vector or matrix with K components, it is assumed that the first K_1 components correspond to the latent topics and the next K_2 components to the supervised topics. Λ_n is a diagonal binary matrix of dimension $K \times K$. The k^{th} diagonal entry is unity if either $1 \leq k \leq K_1$ or $K_1 < k \leq K$ and the n^{th} document is tagged with the k^{th} topic. Also, $\alpha = (\alpha_1, \alpha_2)$ where α_1 is a parameter of a Dirichlet distribution of dimension K_1 and α_2 is a parameter of a Dirichlet distribution of dimension K_2 .

For the m^{th} word in the n^{th} document, sample a topic $z_{nm} \sim \text{multinomial}(\theta'_n)$, where $\theta'_n = (1 - \epsilon)\{\theta_{nk}\}_{k=1}^{K_1} \epsilon\{\Lambda_{n,kk}\theta_{nk}\}_{k=1+K_1}^K$. This implies that the supervised topics are weighted by ϵ and the latent topics are weighted by $(1 - \epsilon)$. Sample the word $w_{nm} \sim \text{multinomial}(\beta_{z_{nm}})$, where β_k is a multinomial distribution over the vocabulary of words corresponding to the k^{th} topic.

For the n^{th} document, generate $Y_n = \arg \max_y \mathbf{r}_y^T \mathbb{E}(\bar{\mathbf{z}}_n)$ where Y_n is the class label associated with the n^{th} document, $\bar{\mathbf{z}}_n = \sum_{m=1}^{M_n} \mathbf{z}_{nm}/M_n$. Here, \mathbf{z}_{nm} is an indicator vector of dimension K . \mathbf{r}_y is a K -dimensional real vector corresponding to the y^{th} class, and it is assumed to have a prior distribution $\mathcal{N}(0, 1/C)$. M_n is the number of words in the n^{th} document. The maximization problem to generate Y_n (or the classification problem) is carried out using a max-margin principle.

Note that predicting each class is effectively treated as a separate task, and that the shared topics are useful for generalizing the performance of the model across classes. In particular, when all classes have few training examples, knowledge transfer between classes can occur through the shared topics. So, the mapping from the original feature space to the topic space is effectively learned using examples from all classes, and a few examples from each class are sufficient to learn the mapping from the reduced topic space to the class labels (see Fig. 3.10). The corresponding graphical model is shown in Fig. 3.9.

3.3.3. Inference and learning

Let us denote the hidden variables by $Z = \{\{z_{nm}\}, \{\theta_n\}\}$, the observed variables by $X = \{w_{nm}\}$ and the model parameters by κ_0 . The joint distribution of the

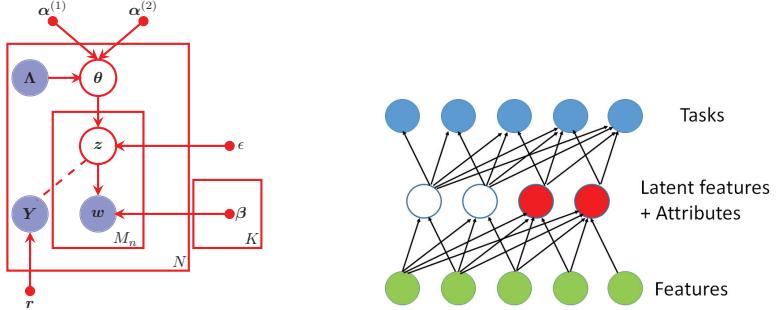


Fig. 3.9. Graphical Model of DSLDA.

Fig. 3.10. Illustration of DSLDA.

hidden and observed variables is:

$$p(\mathbf{X}, \mathbf{Z} | \kappa_0) = \prod_{n=1}^N p(\boldsymbol{\theta}_n | \alpha_n) \prod_{m=1}^{M_n} p(z_{nm} | \boldsymbol{\theta}'_n) p(w_{nm} | \beta_{z_{nm}}). \quad (3.4)$$

To avoid computational intractability, inference and estimation are performed using Variational EM. The factorized approximation to the posterior distribution on hidden variables \mathbf{Z} is given by:

$$q(\mathbf{Z} | \{\kappa_n\}_{n=1}^N) = \prod_{n=1}^N q(\boldsymbol{\theta}_n | \gamma_n) \prod_{m=1}^{M_n} q(z_{nm} | \phi_{nm}), \quad (3.5)$$

where $\boldsymbol{\theta}_n \sim \text{Dir}(\gamma_n) \forall n \in \{1, 2, \dots, N\}$, $z_{nm} \sim \text{multinomial}(\phi_{nm}) \forall n \in \{1, 2, \dots, N\}$ and $\forall m \in \{1, 2, \dots, M_n\}$, and $\kappa_n = \{\gamma_n, \{\phi_{nm}\}\}$, which is the set of variational parameters corresponding to the n^{th} instance. Further, $\gamma_n = (\gamma_{nk})_{k=1}^K \forall n$, and $\phi_{nm} = (\phi_{nmk})_{k=1}^K \forall n, m$. With the use of the lower bound obtained by the factorized approximation, followed by Jensen's inequality, DSLDA reduces to solving the following optimization problem^a:

$$\begin{aligned} & \min_{q, \kappa_0, \{\xi_n\}} \frac{1}{2} \|\mathbf{r}\|^2 - \mathcal{L}(q(\mathbf{Z})) + C \sum_{n=1}^N \xi_n, \\ & \text{s.t. } \forall n, y \neq Y_n : \mathbb{E}[\mathbf{r}^T \Delta f_n(y)] \geq 1 - \xi_n; \xi_n \geq 0. \end{aligned} \quad (3.6)$$

Here, $\Delta f_n(y) = f(Y_n, \bar{\mathbf{z}}_n) - f(y, \bar{\mathbf{z}}_n)$ and $\{\xi_n\}_{n=1}^N$ are the slack variables, and $f(y, \bar{\mathbf{z}}_n)$ is a feature vector whose components from $(y-1)K+1$ to yK are those of the vector $\bar{\mathbf{z}}_n$ and all the others are 0. $\mathbb{E}[\mathbf{r}^T \Delta f_n(y)]$ is the “expected margin” over which the true label Y_n is preferred over a prediction y . From this viewpoint, DSLDA projects the documents onto a combined topic space and then

^aPlease see [11] for further details.

uses a max-margin approach to predict the class label. The parameter C penalizes the margin violation of the training data.

$$\begin{aligned} \phi_{nmk}^* &\propto \Lambda_{n,kk} \exp \left[\psi(\gamma_{nk}) + \log(\beta_{kw_{nm}}) + \log(\epsilon') \right. \\ &\quad \left. + 1/M_n \sum_{y \neq Y_n} \mu_n(y) \mathbb{E}[r_{Y_n k} - r_{yk}] \right] \forall n, m, k. \end{aligned} \quad (3.7)$$

$$\gamma_{nk}^* = \Lambda_{n,kk} \left[\alpha_k + \sum_{m=1}^{M_n} \phi_{nmk} \right] \forall n, k. \quad (3.8)$$

$$\beta_{kv}^* \propto \sum_{n=1}^N \sum_{m=1}^{M_n} \phi_{nmk} \mathbb{I}_{\{w_{nm}=v\}} \forall k, v. \quad (3.9)$$

$$\begin{aligned} \mathcal{L}_{[\alpha_1/\alpha_2]} &= \left[\sum_{n=1}^N \log \left(\Gamma \left(\sum_{k=1}^K \alpha_{nk} \right) \right) - \sum_{n=1}^N \sum_{k=1}^K \log \left(\Gamma(\alpha_{nk}) \right) \right] \\ &\quad + \sum_{n=1}^N \sum_{k=1}^K \left[\psi(\gamma_{nk}) - \psi \left(\sum_{k=1}^K \gamma_{nk} \right) \right] (\alpha_{nk} - 1). \end{aligned} \quad (3.10)$$

Let \mathcal{Q} be the set of all distributions having a fully factorized form as given in (3.5). Let the distribution q^* from the set \mathcal{Q} optimize the objective in Eq. (3.6). The optimal values of corresponding variational parameters are given in Eqs. (3.7) and (3.8). In Eq. (3.7), $\epsilon' = (1 - \epsilon)$ if $k \leq K_1$ and $\epsilon' = \epsilon$ otherwise. Since ϕ_{nm} is a multinomial distribution, the updated values of the K components should be normalized to unity. The optimal values of ϕ_{nm} depend on γ_n and vice-versa. Therefore, iterative optimization is adopted to maximize the lower bound until convergence is achieved.

During testing, one does not observe a document's supervised topics and, in principle, has to explore 2^{K_2} possible combinations of supervised tags – an expensive process. A simple approximate solution, as employed in LLDA [8], is to assume the absence of the variables $\{\Lambda_n\}$ altogether in the test phase, and just treat the problem as inference in MedLDA with K latent topics. One can then threshold over the last K_2 topics if the tags of a test document need to be inferred. Equivalently, one can also assume Λ_n to be an identity matrix of dimension $K \times K \forall n$. This representation ensures that the expressions for update Equations (3.7) and (3.8) do not change in the test phase.

In the M step, the objective in Eq. (3.6) is maximized w.r.t κ_0 . The optimal value of β_{kv} is given in Eq. (3.9). Since β_k is a multinomial distribution, the updated values of the V components should be normalized. However, numerical methods for optimization are required to update α_1 or α_2 . The part of the objective function that depends on α_1 and α_2 is given in Eq. (3.10). The update for the parameter r is carried out using a multi-class SVM solver [47]. With all other model and variational parameters held fixed (*i.e.* with $\mathcal{L}(q)$ held constant), the objective in Eq. (3.6) is optimized w.r.t. r . A reader familiar with the updates in unsupervised LDA can see the subtle (but non-trivial) changes in the update equations for DSLDA.

3.3.4. Non-parametric DSLDA

We now propose a non-parametric extension of DSLDA (NP-DSLDA) that solves the model selection problem and automatically determines the best number of latent topics for modeling the given data. A modified stick breaking construction of Hierarchical Dirichlet Process (HDP) [48], recently introduced in [49] is used here which makes variational inference feasible. The idea in such representation is to share the corpus level atoms across documents by sampling atoms with replacement for each document and modifying the weights of these samples according to some other GEM distribution [48] whose parameter does not depend on the weights of the corpus-level atoms.

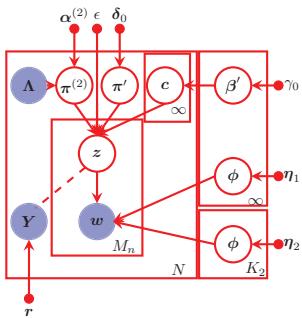


Fig. 3.11. Graphical Model of NP-DSLDA.

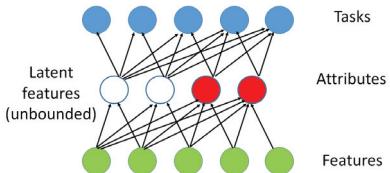


Fig. 3.12. Illustration of NP-DSLDA.

The combination of an infinite number of latent topics with a finite number of supervised topics in a single framework is not trivial and ours is the first model to accomplish this. One simpler solution is to introduce one extra binary hidden variable for each word in each document which could select either the set of la-

tent topics or the set of supervised topics. Subsequently, a word in a document can be sampled from either the supervised or the latent topics based on the value sampled by the hidden “switching” variable. However, the introduction of such extra hidden variables adversely affects model performance as explained in [50]. In NP-DSLDA, we are able to avoid such extra hidden variables by careful modeling of the HDP. This will be evident in the generative process of NP-DSLDA presented below:

- Sample $\phi_{k_1} \sim \text{Dir}(\boldsymbol{\eta}_1) \forall k_1 \in \{1, 2, \dots, \infty\}$ and $\phi_{k_2} \sim \text{Dir}(\boldsymbol{\eta}_2) \forall k_2 \in \{1, 2, \dots, K_2\}$. $\boldsymbol{\eta}_1, \boldsymbol{\eta}_2$ are the parameters of Dirichlet distribution of dimension V .
- Sample $\beta'_{k_1} \sim \text{Beta}(1, \delta_0) \forall k_1 \in \{1, 2, \dots, \infty\}$.
- For the n^{th} document, sample $\boldsymbol{\pi}_n^{(2)} \sim \text{Dir}(\boldsymbol{\Lambda}_n \boldsymbol{\alpha}_2)$. $\boldsymbol{\alpha}_2$ is the parameter of Dirichlet of dimension K_2 . $\boldsymbol{\Lambda}_n$ is a diagonal binary matrix of dimension $K_2 \times K_2$. The k^{th} diagonal entry is unity if the n^{th} word is tagged with the k^{th} supervised topic.
- $\forall n, \forall t \in \{1, 2, \dots, \infty\}$, sample $\pi'_{nt} \sim \text{Beta}(1, \alpha_0)$. Assume $\boldsymbol{\pi}_n^{(1)} = (\pi_{nt})_t$ where $\pi_{nt} = \pi'_{nt} \prod_{l < t} (1 - \pi'_{nl})$.
- $\forall n, \forall t$, sample $c_{nt} \sim \text{multinomial}(\boldsymbol{\beta})$ where $\beta_{k_1} = \beta'_{k_1} \prod_{l < k_1} (1 - \beta'_l)$. $\boldsymbol{\pi}_n^{(1)}$ represents the probability of selecting the sampled atoms in c_n . Due to sampling with replacement, c_n can contain multiple atoms of the same index from the corpus level DP.
- For the m^{th} word in the n^{th} document, sample $z_{nm} \sim \text{multinomial}((1 - \epsilon)\boldsymbol{\pi}_n^{(1)}, \epsilon\boldsymbol{\pi}_n^{(2)})$. This implies that w.p. ϵ , a topic is selected from the set of supervised topics and w.p. $(1 - \epsilon)$, a topic is chosen from the set of (infinite number of) unsupervised topics. Note that by weighting the $\boldsymbol{\pi}$'s appropriately, the need for additional hidden “switching” variable is avoided.
- Sample w_{nm} from a multinomial given by the following equation:

$$\prod_{k_1=1}^{\infty} \prod_{v=1}^V \phi_{k_1 v}^{\mathbb{I}_{\{w_{nm}=v\}} \mathbb{I}_{\{c_{nz_{nm}}=k_1 \in \{1, \dots, \infty\}\}}} \prod_{k_2=1}^{K_2} \prod_{v=1}^V \phi_{k_2 v}^{\mathbb{I}_{\{w_{nm}=v\}} \mathbb{I}_{\{z_{nm}=k_2 \in \{1, \dots, K_2\}\}}} \quad (3.11)$$

The corresponding graphical model is shown in Fig. 3.11. The joint distribution of NP-DSLDA is given as follows:

$$\begin{aligned}
 p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\kappa}_0) = & \prod_{k_1=1}^{\infty} p(\phi_{k_1} | \boldsymbol{\eta}_1) p(\beta'_{k_1} | \boldsymbol{\delta}_0) \prod_{k_2=1}^{K_2} p(\phi_{k_2} | \boldsymbol{\eta}_2) \prod_{n=1}^N p(\boldsymbol{\pi}_n^{(2)} | \boldsymbol{\alpha}_2) \\
 & \prod_{t=1}^{\infty} p(\boldsymbol{\pi}'^{(1)}_{nt} | \boldsymbol{\alpha}_0) p(c_{nt} | \boldsymbol{\beta}') \prod_{m=1}^{M_n} p(z_{nm} | \boldsymbol{\pi}_n^{(1)}, \boldsymbol{\pi}_n^{(2)}, \epsilon) p(w_{nm} | \boldsymbol{\phi}, c_{nz_{nm}}, z_{nm}).
 \end{aligned} \tag{3.12}$$

As an approximation to the posterior distribution over the hidden variables, we use the following factorized distribution:

$$\begin{aligned}
 q(\mathbf{Z} | \boldsymbol{\kappa}) = & \prod_{k_1=1}^{\bar{K}_1} q(\phi_{k_1} | \boldsymbol{\lambda}_{k_1}) \prod_{k_2=1}^{K_2} q(\phi_{k_2} | \boldsymbol{\lambda}_{k_2}) \prod_{k_1=1}^{\bar{K}_1-1} q(\beta'_{k_1} | u_{k_1}, v_{k_1}) \\
 & \prod_{n=1}^N q(\boldsymbol{\pi}_n^{(2)} | \boldsymbol{\gamma}_n) \prod_{t=1}^{T-1} q(\boldsymbol{\pi}'^{(1)}_{nt} | a_{nt}, b_{nt}) \prod_{t=1}^T q(c_{nt} | \boldsymbol{\varphi}_{nt}) \prod_{m=1}^{M_n} q(z_{nm} | \boldsymbol{\zeta}_{nm}).
 \end{aligned} \tag{3.13}$$

Here, $\boldsymbol{\kappa}_0$ and $\boldsymbol{\kappa}$ denote the sets of model and variational parameters, respectively. \bar{K}_1 is the truncation limit of the corpus-level Dirichlet Process and T is the truncation limit of the document-level Dirichlet Process. $\{\boldsymbol{\lambda}_k\}$ are the parameters of Dirichlet each of dimension V . $\{u_{k_1}, v_{k_1}\}$ and $\{a_{nt}, b_{nt}\}$ are the parameters of variational Beta distribution corresponding to corpus level and document level sticks respectively. $\{\boldsymbol{\varphi}_{nt}\}$ are multinomial parameters of dimension \bar{K}_1 and $\{\boldsymbol{\zeta}_{nm}\}$ are multinomials of dimension $(T + K_2)$. $\{\boldsymbol{\gamma}_n\}_n$ are parameters of Dirichlet distribution of dimension K_2 .

The underlying optimization problem takes the same form as in Eq. (3.6). The only difference lies in the calculation of $\Delta f_n(y) = f(Y_n, \bar{s}_n) - f(y, \bar{s}_n)$. The first set of dimensions of \bar{s}_n (corresponding to the unsupervised topics) is given by $1/M_n \sum_{m=1}^{M_n} \mathbf{c}_{nz_{nm}}$, where \mathbf{c}_{nt} is an indicator vector over the set of unsupervised topics. The following K_2 dimensions (corresponding to the supervised topics) are given by $1/M_n \sum_{m=1}^{M_n} \mathbf{z}_{nm}$. After the variational approximation with \bar{K}_1 number of corpus level sticks, \bar{s}_n turns out to be of dimension $(\bar{K}_1 + K_2)$ and the feature vector $f(y, \bar{s}_n)$ constitutes $Y(\bar{K}_1 + K_2)$ elements. The components of $f(y, \bar{s}_n)$ from $(y-1)(\bar{K}_1 + K_2) + 1$ to $y(\bar{K}_1 + K_2)$ are those of the vector \bar{s}_n and all the others are 0. Essentially, due to the variational approximation, NP-DSLDA projects each document on to a combined topic space of dimension $(\bar{K}_1 + K_2)$ and

learns the mapping from this space to the classes.

$$\begin{aligned} \zeta_{nmt}^* \propto & \exp \left[[\psi(a_{nt}) - \psi(a_{nt} + b_{nt})] \mathbb{I}_{\{t < T\}} + \sum_{t'=1}^{t-1} [\psi(b_{nt'}) - \psi(a_{nt'} + b_{nt'})] \right. \\ & + \sum_{k_1=1}^{\bar{K}_1} \varphi_{ntk_1} \left[\psi(\lambda_{k_1 w_{nm}}) - \psi \left(\sum_{v=1}^V \lambda_{k_1 v} \right) \right] \\ & \left. + \sum_{y \neq Y_n} \mu_n(y) \sum_{k_1=1}^{\bar{K}_1} \mathbb{E}[r_{Y_n k_1} - r_{yk_1}] \varphi_{ntk_1} \right] \forall n, m, t. \end{aligned} \quad (3.14)$$

$$\begin{aligned} \zeta_{nm(T+k_2)}^* \propto & \Lambda_{nk_2 k_2} \exp \left[\psi(\gamma_{nk_2}) - \psi \left(\sum_{k_2=1}^{K_2} \gamma_{nk_2} \right) + \psi \left(\lambda_{(\bar{K}_1+k_2) w_{nm}} \right) \right. \\ & \left. - \psi \left(\sum_{v=1}^V \lambda_{(\bar{K}_1+k_2)v} \right) + 1/M_n \sum_{y \neq Y_n} \mu_n(y) \mathbb{E}[r_{Y_n (\bar{K}_1+k_2)} - r_{y (\bar{K}_1+k_2)}] \right] \forall n, m, k_2. \end{aligned} \quad (3.15)$$

$$\begin{aligned} \varphi_{ntk_1}^* \propto & \exp \left[[\psi(u_{k_1}) - \psi(u_{k_1} + v_{k_1})] \mathbb{I}_{\{k_1 < K_1\}} \right. \\ & + \sum_{k'=1}^{k_1-1} [\psi(v_{k'}) - \psi(u_{k'} + v_{k'})] + \sum_{m=1}^{M_n} \zeta_{nmt} \left[\psi(\lambda_{k_1 w_{nm}}) - \psi \left(\sum_{v=1}^V \lambda_{k_1 v} \right) \right] \\ & \left. + 1/M_n \sum_{y \neq Y_n} \mu_n(y) \mathbb{E}[r_{Y_n k_1} - r_{yk_1}] \left(\sum_{m=1}^{M_n} \zeta_{nmt} \right) \right] \forall n, t, k_1. \end{aligned} \quad (3.16)$$

Some of the update equations of NP-DSLDA are given in the above equations, where $\{\varphi_{ntk_1}\}$ are the set of variational parameters that characterize the assignment of the documents to the global set of $(K_1 + K_2)$ topics. One can see how the effect of the class labels is included in the update equation of $\{\varphi_{ntk_1}\}$ via the average value of the parameters $\{\zeta_{nmt}\}$. This follows intuitively from the generative assumption. update exists for the model parameters and hence numerical optimization has to be used. Other updates are either similar to DSLDA or the model in [49] and are omitted due to space constraints. $\{\zeta_{nm}\}$, corresponding to supervised and unsupervised topics, should be individually normalized and then scaled by ϵ and $(1 - \epsilon)$ respectively. Otherwise, the effect of the Dirichlet prior

on supervised topics will get compared to that of the GEM prior on the unsupervised topics which does not follow the generative assumptions. The variational parameters $\{\lambda_k\}$ and $\{\varphi_{nt}\}$ are also normalized.

Note that NP-DSLDA offers some flexibility with respect to the latent topics that could be dominant for a specific task. One could therefore postulate that NP-DSLDA can learn the clustering of tasks from the data itself by making a subset of latent topics to be dominant for a set of tasks. Although do not have supporting experiments, NP-DSLDA is capable in principle of performing clustered multi-task learning without any prior assumption on the relatedness of the tasks.

3.4. Active Multitask Learning using Both Supervised and Shared Latent Topics

3.4.1. Active Doubly Supervised Latent Dirichlet Allocation (Act-DSLDA)

We will treat examples as “documents” which consist of a “bag of words” for text or a “bag of visual words” for images. Assume we are given an initial training corpus \mathcal{L} with N documents belonging to Y different classes. Further assume that each of these training documents is also annotated with a set of K_2 different “supervised topics”. The objective is to train a model using the words in a document, as well as the associated supervised topics and class labels, and then use this model to classify completely unlabeled test documents for which no topics or class labels are provided.

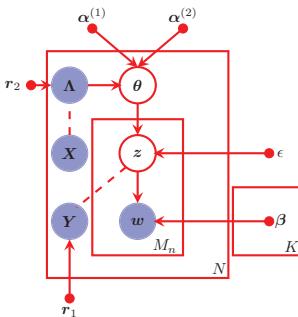


Fig. 3.13. Graphical Model of Act-DSLDA.

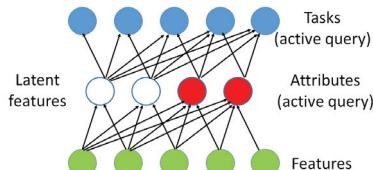


Fig. 3.14. Illustration of Act-DSLDA.

When the learning starts, \mathcal{L} is assumed to have fully labeled documents. However, as the learning progresses more documents are added to the pool \mathcal{L} with

class and/or a subset of supervised topics labeled. Therefore, at any intermediate point of the learning process, \mathcal{L} can be assumed to contain several sets: $\mathcal{L} = \{\mathcal{T} \cup \mathcal{T}_C \cup \mathcal{T}_{A_1} \cup \mathcal{T}_{A_2} \cup \dots \cup \mathcal{T}_{A_{K_2}}\}$, where \mathcal{T} contains fully labeled documents (*i.e.* with class and all supervised topics labeled), \mathcal{T}_C are the documents that have class labels, and $1 \leq k \leq K_2$, \mathcal{T}_{A_k} are the documents that have the k^{th} supervised topic labeled. Since, human-provided labels are expensive to obtain, we design an active learning framework where the model can query over an unlabeled pool \mathcal{U} and request either class labels or a subset of the supervised topics.

Please note that the proposed frameworks support general MTL; however, our datasets, as explained in Section 3.5, happen to be multiclass, where each class is treated as a separate “task” (as typical in multi-class learning based on binary classifiers). However, the frameworks are not in any way restricted to multiclass MTL. The Act-DSLDA generative model is defined as follows. For the n^{th} document, sample a topic selection probability vector $\theta_n \sim \text{Dir}(\alpha_n)$, where $\alpha_n = \Lambda_n \alpha$ and α is the parameter of a Dirichlet distribution of dimension K , the total number of topics. The topics are assumed to be of two types – latent and supervised, and there are K_1 latent topics and K_2 supervised topics ($K = K_1 + K_2$). Latent topics are never observed, while supervised topics are observed in the training data but not in the test data. Henceforth, in each vector or matrix with K components, it is assumed that the first K_1 components correspond to the latent topics and the next K_2 components to the supervised topics. Λ_n is a diagonal binary matrix of dimension $K \times K$. The k^{th} diagonal entry is unity if *either* $1 \leq k \leq K_1$ *or* $K_1 < k \leq K$ and the n^{th} document is tagged with the k^{th} topic. Also, $\alpha = (\alpha^{(1)}, \alpha^{(2)})$ where $\alpha^{(1)}$ is a parameter of a Dirichlet distribution of dimension K_1 and $\alpha^{(2)}$ is a parameter of a Dirichlet distribution of dimension K_2 .

In the test data, the supervised topics are not observed and one has to infer them from either the parameters of the model or use some other auxiliary information. Since one of our objectives is to query over the supervised topics as well as the final category, we train a set of binary SVM classifiers that can predict the individual attributes from the features of the data. We denote the parameters of such classifiers by $\{r_{2k}\}_{1 \leq k \leq K_2}$. This is important to get an uncertainty measure over the supervised topics. To further clarify the issue, let us consider that only one supervised topic has to be labeled by the annotator for the n^{th} document from the set of supervised topics of size K_2 . To select the most uncertain topic, one needs to compare the uncertainty of predicting the presence or absence of the individual topics. This uncertainty is different from that calculated from the conditional distribution calculated from the posterior over θ_n .

For the m^{th} word in the n^{th} document, sample a topic $z_{nm} \sim \text{multinomial}(\theta'_n)$, where $\theta'_n = (1 - \epsilon)\{\theta_{nk}\}_{k=1}^{K_1} \epsilon \{\Lambda_{n,kk} \theta_{nk}\}_{k=1+K_1}^K$. This implies that the super-

vised topics are weighted by ϵ and the latent topics are weighted by $(1-\epsilon)$. Sample the word $w_{nm} \sim \text{multinomial}(\beta_{z_{nm}})$, where β_k is a multinomial distribution over the vocabulary of words corresponding to the k^{th} topic.

For the n^{th} document, generate $Y_n = \arg \max_y \mathbf{r}_{1y}^T \mathbb{E}(\bar{\mathbf{z}}_n)$ where Y_n is the class label associated with the n^{th} document, $\bar{\mathbf{z}}_n = \sum_{m=1}^{M_n} \mathbf{z}_{nm}/M_n$. Here, \mathbf{z}_{nm} is an indicator vector of dimension K . \mathbf{r}_{1y} is a K -dimensional real vector corresponding to the y^{th} class, and it is assumed to have a prior distribution $\mathcal{N}(0, 1/C)$. M_n is the number of words in the n^{th} document. The maximization problem to generate Y_n (i.e. the classification problem) is carried out using the max-margin principle and we use online SVMs [45, 46] for such updates. Since the model has to be updated incrementally in the active selection step, a batch SVM solver is not applicable, while an online SVM allows one to update the learned weights incrementally given each new example. Note that predicting each class is treated as a separate task, and that the shared topics are useful for generalizing the performance of the model across classes.

3.4.2. Inference and learning

Inference and parameter estimation have two phases – one for the batch case when the model is trained with fully labeled data, and the other for the active selection step where the model has to be incrementally updated to observe the effect of any labeled information that is queried from the oracle.

3.4.2.1. Learning in Batch Mode

Let us denote the hidden variables by $\mathbf{Z} = \{\{z_{nm}\}, \{\theta_n\}\}$, the observed variables by $\mathbf{X} = \{w_{nm}\}$ and the model parameters by κ_0 . The joint distribution of the hidden and observed variables is:

$$p(\mathbf{X}, \mathbf{Z} | \kappa_0) = \prod_{n=1}^N p(\theta_n | \alpha_n) \prod_{m=1}^{M_n} p(z_{nm} | \theta'_n) p(w_{nm} | \beta_{z_{nm}}). \quad (3.17)$$

To avoid computational intractability, inference and estimation are performed using variational EM. The factorized approximation of the posterior distribution with hidden variables \mathbf{Z} is given by:

$$q(\mathbf{Z} | \{\kappa_n\}_{n=1}^N) = \prod_{n=1}^N q(\theta_n | \gamma_n) \prod_{m=1}^{M_n} q(z_{nm} | \phi_{nm}), \quad (3.18)$$

where $\theta_n \sim \text{Dir}(\gamma_n) \forall n \in \{1, 2, \dots, N\}$, $z_{nm} \sim \text{multinomial}(\phi_{nm}) \forall n \in \{1, 2, \dots, N\}$ and $\forall m \in \{1, 2, \dots, M_n\}$, and $\kappa_n = \{\gamma_n, \{\phi_{nm}\}\}$, which is the set of variational parameters corresponding to the n^{th} instance. Further, $\gamma_n =$

$(\gamma_{nk})_{k=1}^K \forall n$, and $\phi_{nm} = (\phi_{nmk})_{k=1}^K \forall n, m$. With the use of the lower bound obtained by the factorized approximation, followed by Jensen's inequality, Act-DSLDA reduces to solving the following optimization problem^b:

$$\min_{q, \kappa_0, \{\xi_n\}} \frac{1}{2} \|\mathbf{r}_1\|^2 - \mathcal{L}(q(\mathbf{Z})) + C \sum_{n=1}^N \xi_n \mathbb{I}_{\mathcal{T}_C, n},$$

s.t. $\forall n \in \mathcal{T}_C, y \neq Y_n : \mathbb{E}[\mathbf{r}_1^T \Delta f_n(y)] \geq 1 - \xi_n; \xi_n \geq 0.$ (3.19)

Here, $\Delta f_n(y) = f(Y_n, \bar{z}_n) - f(y, \bar{z}_n)$ and $\{\xi_n\}_{n=1}^N$ are the slack variables, and $f(y, \bar{z}_n)$ is a feature vector whose components from $(y-1)K+1$ to yK are those of the vector \bar{z}_n and all the others are 0. $\mathbb{E}[\mathbf{r}_1^T \Delta f_n(y)]$ is the “expected margin” over which the true label Y_n is preferred over a prediction y . From this viewpoint, Act-DSLDA projects the documents onto a combined topic space and then uses a max-margin approach to predict the class label. The parameter C penalizes the margin violation of the training data. The indicator variable $\mathbb{I}_{\mathcal{T}_C, n}$ is unity if the n^{th} document has a class label (*i.e.* $n \in \mathcal{T}_C$) and 0 otherwise. This implies that only the documents that have class labels are used to update the parameters of the online SVM.

Let \mathcal{Q} be the set of all distributions having a fully factorized form as given in (3.18). Note that such a factorized approximation makes the use of incremental variation of EM possible in the active selection step following the discussion in Section 3.2.5. Let the distribution q^* from the set \mathcal{Q} optimize the objective in Eq. (3.19). The optimal values of the corresponding variational parameters are same as those of DSLDA [51]. The optimal values of ϕ_{nm} depend on γ_n and vice-versa. Therefore, iterative optimization is adopted to maximize the lower bound until convergence is achieved.

During testing, one does not observe a document's supervised topics and instead an approximate solution, as also used in [8, 51], is employed where the variables $\{\Lambda_n\}$ are assumed to be absent altogether in the test phase, and the problem is treated as inference in MedLDA with K latent topics.

In the M step, the objective in Eq. (3.19) is maximized w.r.t κ_0 . The optimal value of β_{kv} is again similar to that of DSLDA [51]. However, numerical methods for optimization are required to update α_1 or α_2 . The update for the parameters $\{\mathbf{r}_{1y}\}_{y=1}^Y$ is carried out using online SVM [45, 46] following Eq. (3.19).

3.4.2.2. Incremental Learning in Active Selection

The method of Expected Entropy Reduction requires one to take an example from the unlabeled pool and one of its possible labels, update the model, and observe

^bPlease see [11] for further details.

the generalized error on the unlabeled pool. This process is computationally expensive unless there is an efficient way to update the model incrementally. The incremental view of EM and the online SVM framework are appropriate for such updates.

Consider that a completely unlabeled or partially labeled document, indexed by n' , is to be included in the labeled pool with one of the $(K_2 + 1)$ labels (one for the class label and each different supervised topic), indexed by k' . In the E step, variational parameters corresponding to all other documents except for the n' th one are kept fixed and the variational parameters for only the n' th document are updated. In the M-step, we keep the priors $\{\alpha^{(1)}, \alpha^{(2)}\}$ over the topics and the SVM parameters r_2 fixed as there is no easy way to update such parameters incrementally. From the empirical point of view, these parameters do not change much w.r.t. the variational parameters (or features in topic space representation) of a single document. However, the update of the parameters $\{\beta, r_1\}$ is easier. Updating β is accomplished by a simple update of the sufficient statistics. Updating r_1 is done using the “ProcessNew” operation of online SVM followed by a few iterations of “ProcessOld”. The selection of the document-label pair is guided by the measure given in Eq. (3.1). Note that since SVM uses hinge loss which, in turn, upper bounds the 0–1 loss in classification, use of the measure from Eq. (3.1) for active query selection is justified.

From the modeling perspective, the difference between DSLDA [51] and Act-DSLDA lies in maintaining attribute classifiers and ignoring documents in the max-margin learning that do not have any class label. Online SVM for max-margin learning is essential in the batch mode just to maintain the support vectors and incrementally update them in the active selection step. One could also use incremental EM for batch mode training. However, that is computationally more complex when the labeled dataset is large, as the E step for each document is followed by an M-step in incremental EM.

3.4.3. Active Non-parametric DSLDA (Act-NPDSLDA)

A non-parametric extension of Act-DSLDA (Act-NPDSLDA) automatically determines the best number of latent topics for modeling the given data. It uses a modified stick breaking construction of Hierarchical Dirichlet Process (HDP), recently introduced in [49], to make variational inference feasible. The Act-NPDSLDA generative model is presented below.

- Sample $\phi_{k_1} \sim \text{Dir}(\eta_1) \forall k_1 \in \{1, 2, \dots, \infty\}$ and $\phi_{k_2} \sim \text{Dir}(\eta_2) \forall k_2 \in \{1, 2, \dots, K_2\}$. η_1, η_2 are the parameters of Dirichlet distribution of dimension V . Also, sample $\beta'_{k_1} \sim \text{Beta}(1, \delta_0) \forall k_1 \in \{1, 2, \dots, \infty\}$.

- For the n^{th} document, sample $\pi_n^{(2)} \sim \text{Dir}(\Lambda_n \alpha^{(2)})$. $\alpha^{(2)}$ is the parameter of Dirichlet of dimension K_2 . Λ_n is a diagonal binary matrix of dimension $K_2 \times K_2$. The k^{th} diagonal entry is unity if the n^{th} word is tagged with the k^{th} supervised topic. Similar to the case of Act-DSLDA, in the test data, the supervised topics are not observed and the set of binary SVM classifiers, trained with document-attribute pair data, are used to predict the individual attributes from the input features. The parameters of such classifiers are denoted by $\{r_{2k}\}_{1 \leq k \leq K_2}$.
- $\forall n, \forall t \in \{1, 2, \dots, \infty\}$, sample $\pi'_{nt} \sim \text{Beta}(1, \alpha_0)$. Assume $\pi_n^{(1)} = (\pi_{nt})_t$ where $\pi_{nt} = \pi'_{nt} \prod_{l < t} (1 - \pi'_{nl})$. $\forall n, \forall t$, sample $c_{nt} \sim \text{multinomial}(\beta)$ where $\beta_{k1} = \beta'_{k1} \prod_{l < k_1} (1 - \beta'_l)$. $\pi_n^{(1)}$ represents the probability of selecting the sampled atoms in c_n .
- For the m^{th} word in the n^{th} document, sample $z_{nm} \sim \text{multinomial}((1 - \epsilon)\pi_n^{(1)}, \epsilon\pi_n^{(2)})$. This implies that with probability ϵ , a topic is selected from the set of supervised topics and with probability $(1 - \epsilon)$, a topic is chosen from the set of unsupervised topics. Sample w_{nm} from a multinomial given by Eq. (3.3).

- For the n^{th} document, generate $Y_n = \arg \max_y r_{1y}^T \mathbb{E}(\bar{z}_n)$ where Y_n is the class label associated with the n^{th} document, $\bar{z}_n = \sum_{m=1}^{M_n} z_{nm}/M_n$. The maximization problem to generate Y_n (i.e. the classification problem) is carried out using an online support vector machine. The joint distribution of the hidden and observed variables is given in Eq. (3.1).

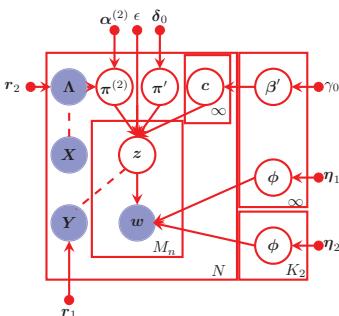


Fig. 3.15. Graphical Model of Act-NPDSLDA.

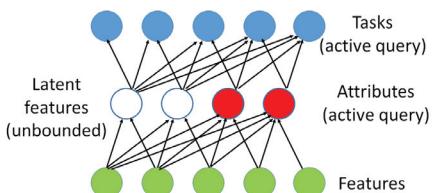


Fig. 3.16. Illustration of Act-NPDSLDA.

Table 3.1. Distributions in Act-NPDSLDA.

Joint Distribution of Act-NPDSLDA
$p(\mathbf{X}, \mathbf{Z} \kappa_0) = \prod_{k_1=1}^{\infty} p(\phi_{k_1} \eta_1) p(\beta'_{k_1} \delta_0) \prod_{k_2=1}^{K_2} p(\phi_{k_2} \eta_2) \prod_{n=1}^N p(\pi_n^{(2)} \alpha_2) \prod_{l=1}^{\infty} p(\pi_{nl}^{(1)} \alpha_0) p(c_{nt} \beta') \prod_{m=1}^{M_n} p(z_{nm} \pi_n^{(1)}, \pi_n^{(2)}, \epsilon) p(w_{nm} \phi, c_{nz_{nm}}, z_{nm}). \quad (1)$
Variational Distribution of Act-NPDSLDA
$q(\mathbf{Z} \kappa) = \prod_{k_1=1}^{K_1} q(\phi_{k_1} \lambda_{k_1}) \prod_{k_2=1}^{K_2} q(\phi_{k_2} \lambda_{k_2}) \prod_{k_1=1}^{K_1-1} q(\beta'_{k_1} u_{k_1}, v_{k_1}) \prod_{n=1}^N q(\pi_n^{(2)} \gamma_n) \prod_{t=1}^{T-1} q(\pi_{nt}^{(1)} a_{nt}, b_{nt}) \prod_{t=1}^T q(c_{nt} \varphi_{nt}) \prod_{m=1}^{M_n} q(z_{nm} \zeta_{nm}). \quad (2)$
Multinomial Distribution for Sampling Words in Act-NPDSLDA
$\prod_{k_1=1}^{\infty} \prod_{v=1}^V \phi_{k_1 v}^{\mathbb{I}\{w_{nm}=v\}} \mathbb{I}\{c_{nz_{nm}}=k_1 \in \{1, \dots, \infty\}\} \prod_{k_2=1}^{K_2} \prod_{v=1}^V \phi_{k_2 v}^{\mathbb{I}\{w_{nm}=v\}} \mathbb{I}\{z_{nm}=k_2 \in \{1, \dots, K_2\}\}. \quad (3)$

3.4.4. Inference and Learning

3.4.4.1. Learning in Batch Mode

As an approximation to the posterior distribution over the hidden variables, we use the factorized distribution given in Eq. (3.2). κ_0 and κ denote the sets of model and variational parameters, respectively. \bar{K}_1 is the truncation limit of the corpus-level Dirichlet Process and T is the truncation limit of the document-level Dirichlet Process. $\{\lambda_k\}$ are the parameters of the Dirichlet, each of dimension V . $\{u_{k_1}, v_{k_1}\}$ and $\{a_{nt}, b_{nt}\}$ are the parameters of Beta distribution corresponding to corpus level and document level sticks respectively. $\{\varphi_{nt}\}$ are multinomial parameters of dimension \bar{K}_1 and $\{\zeta_{nm}\}$ are multinomials of dimension $(T + K_2)$. $\{\gamma_n\}_n$ are parameters of the Dirichlet distribution of dimension K_2 .

The underlying optimization problem takes the same form as in Eq. (3.19). The only difference lies in the calculation of $\Delta f_n(y) = f(Y_n, \bar{s}_n) - f(y, \bar{s}_n)$. The first set of dimensions of \bar{s}_n (corresponding to the unsupervised topics) is given by $1/M_n \sum_{m=1}^{M_n} c_{nz_{nm}}$, where c_{nt} is an indicator vector over the set of unsupervised topics. The following K_2 dimensions (corresponding to the supervised topics) are given by $1/M_n \sum_{m=1}^{M_n} z_{nm}$. After the variational approximation with \bar{K}_1 number of corpus level sticks, \bar{s}_n turns out to be of dimension $(\bar{K}_1 + K_2)$ and the feature vector $f(y, \bar{s}_n)$ constitutes $Y(\bar{K}_1 + K_2)$ elements. The components of $f(y, \bar{s}_n)$ from $(y-1)(\bar{K}_1 + K_2) + 1$ to $y(\bar{K}_1 + K_2)$ are those of the vector \bar{s}_n and all the others are 0. The E-step update equations of Act-NPDSLDA are similar to NP-DSLDA [51]. The M-step updates are similar to Act-DSLDA and are omitted here due to space constraints.

3.4.4.2. Incremental Learning in Active Selection

Assume that a completely unlabeled or partially labeled document, indexed by n' , is to be included in the labeled pool with the k' th label. In the E step, variational

parameters corresponding to all other documents except for the n' th one is kept fixed and the variational parameters for only the n' th document are updated. The incremental update of the “global” variational parameters $\{u_{k_1}, v_{k_1}\}_{k_1=1}^{K_1}$ is also straightforward following the equations given in [51]. In the M-step, we keep the priors $\{\eta_1, \eta_2, \alpha^{(2)}\}$ and the SVM parameters r_2 fixed but the parameters r_1 are updated using online SVM.

3.5. Experimental Evaluation

3.5.1. Data description

Our evaluation used two datasets, a text corpus and a multi-class image database, as described below.

3.5.1.1. aYahoo data

The first set of experiments was conducted with the aYahoo image dataset from [2] which has 12 classes – carriage, centaur, bag, building, donkey, goat, jetski, monkey, mug, statue, wolf, and zebra.^c Each image is annotated with relevant visual attributes such as “has head”, “has wheel”, “has torso” and 61 others, which we use as the supervised topics. Using such intermediate “attributes” to aid visual classification has become a popular approach in computer vision [3, 4]. After extracting SIFT features [52] from the raw images, quantization into 250 clusters is performed, defining the vocabulary for the bag of visual words. Images with less than two attributes were discarded. The resulting dataset of size 2,275 was equally split into training and test data.

3.5.1.2. ACM Conference data

The text corpus consists of conference chapter abstracts from two groups of conferences. The first group has four conferences related to data mining – WWW, SIGIR, KDD, and ICML, and the second group consists of two VLSI conferences – ISPD and DAC. The classification task is to determine the conference at which the abstract was published. As supervised topics, we use keywords provided by the authors, which are presumably useful in determining the conference venue. Since authors usually take great care in choosing keywords so that their chapter is retrieved by relevant searches, we believed that such keywords made a good choice of supervised topics. Part of the data, crawled from ACM’s website, was used in [53]. A total of 2,300 abstracts were collected each of which had at least

^c<http://vision.cs.uiuc.edu/attributes/>

three keywords and an average of 78 (± 33.5) words. After stop-word removal, the vocabulary size for the assembled data is 13,412 words. The final number of supervised topics, after some standard pre-processing of keywords, is 55. The resulting dataset was equally split into training and test data.

3.5.2. Methodology for experiments with Multitask Learning

In order to demonstrate the contribution of each aspect of the overall model, DSLDA and NP-DSLDA are compared against the following simplified models:

- MedLDA with **one-vs-all** classification (MedLDA-OVA) (shown in Fig. 3.17): A separate model is trained for each class using a one-vs-all approach leaving no possibility of transfer across classes.
- MedLDA with **multitask learning** (MedLDA-MTL) (shown in Fig. 3.18): A single model is learned for all classes where the latent topics are shared across classes.
- DSLDA with **only shared supervised topics** (DSLDA-OSST) (shown in Fig. 3.19): A model in which supervised topics are used and shared across classes but there are no latent topics.
- DSLDA with **no shared latent topics** (DSLDA-NSLT) (shown in Fig. 3.20): A model in which only supervised topics are shared across classes and a separate set of latent topics is maintained for each class.
- **Majority class method (MCM)**: A simple baseline which always picks the most common class in the training data.

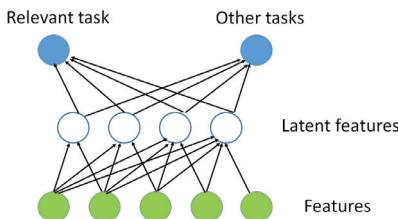


Fig. 3.17. Illustration of MedLDA-OVA.

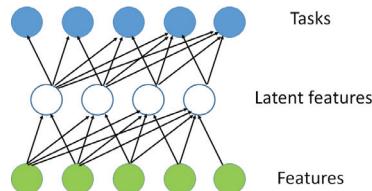


Fig. 3.18. Illustration of MedLDA-MTL.

These baselines are useful for demonstrating the utility of *both* supervised and latent shared topics for multitask learning in DSLDA. MedLDA-OVA is a non-transfer method, where a separate model is learned for each of the classes, *i.e.* one of the many classes is considered as the positive class and the union of the remaining ones is treated as the negative class. Since the models for each class are

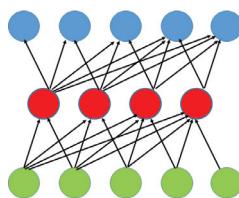


Fig. 3.19. Illustration of DSLDA-OSST.

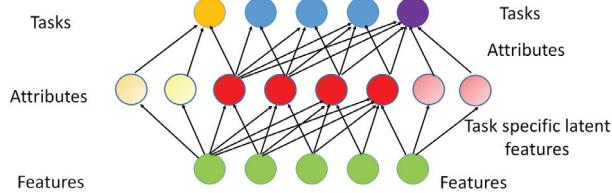
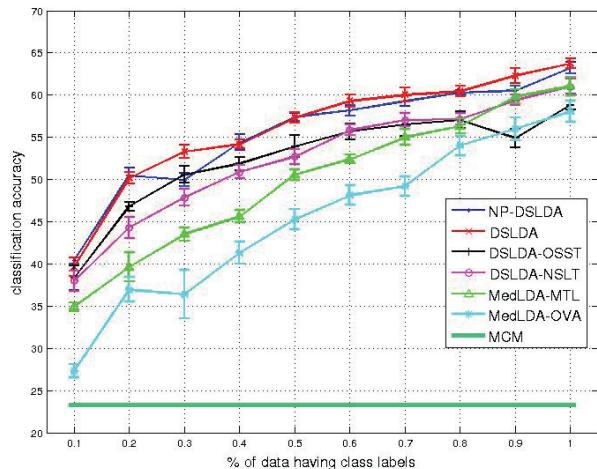
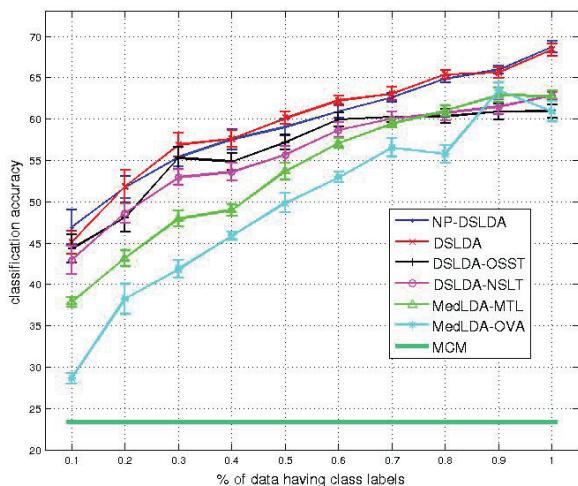


Fig. 3.20. Illustration of DSLDA-NSLT.

trained separately, there is no possibility of sharing inductive information across classes. MedLDA-MTL trains on examples from all classes simultaneously, and thus allows for sharing of inductive information *only* through a common set of latent topics. In DSLDA-OSST, only supervised topics are maintained and knowledge transfer can *only* take place via these supervised topics. DSLDA-NSLT uses shared supervised topics but also includes latent topics which are *not* shared across classes. This model provides for transfer *only* through shared supervised topics but provides extra modeling capacity compared to DSLDA-OSST through the use of latent topics that are not shared. DSLDA and NP-DSLDA are MTL frameworks where both supervised *and* latent topics are shared across all classes. Note that, all of the baselines can be implemented using DSLDA with a proper choice of Λ and ϵ . For example, DSLDA-OSST is just a special case of DSLDA with ϵ fixed at 1.

In order to explore the effect of different amounts of both types of supervision, we varied the amount of both topic-level and class-level supervision. Specifically, we provided topic supervision for a fraction, p_1 , of the overall training set, and then provided class supervision for only a further fraction p_2 of this data. Therefore, only $p_1 * p_2$ of the overall training data has class supervision. By varying the number of latent topics from 20 to 200 in steps of 10, we found that $K_1 = 100$ generally worked the best for all the parametric models. Therefore, we show parametric results for 100 latent topics. For each combination of (p_1, p_2) , 50 random trials were performed with $C = 10$. To maintain equal representational capacity, the total number of topics K is held the same across all parametric models (except for DSLDA-OSST where the total number of topics is K_2). For NP-DSLDA, following the suggestion of [49], we set $\bar{K}_1 = 150$ and $T = 40$, which produced uniformly good results. When required, ϵ was chosen using 5-fold internal cross-validation using the training data.

Fig. 3.21. $p_1 = 0.5$ (aYahoo).Fig. 3.22. $p_1 = 0.7$ (aYahoo).

3.5.3. Multitask Learning results

Figures 3.23 and 3.24 present representative learning curves for the image data, showing how classification accuracy improves as the amount of class supervision (p_2) is increased. Results are shown for two different amounts of topic supervision ($p_1 = 0.5$ and $p_1 = 0.7$). Figures 3.21 and 3.22 present similar learning curves for the text data. The error bars in the curves show standard deviations across the 50 trials.

The results demonstrate that DSLDA and NP-DSLDA quite consistently outperform all of the baselines, clearly demonstrating the advantage of combining both types of topics. NP-DSLDA performs about as well as DSLDA, for which the optimal number of latent topics has been chosen using an expensive model-selection search. This demonstrates that NP-DSLDA is doing a good job of automatically selecting an appropriate number of latent topics.

Overall, DSLDA-OSST and MedLDA-MTL perform about the same, showing that, individually, both latent and supervised shared topics each support multitask learning about equally well when used alone. However, combining both types of topics provides a clear improvement.

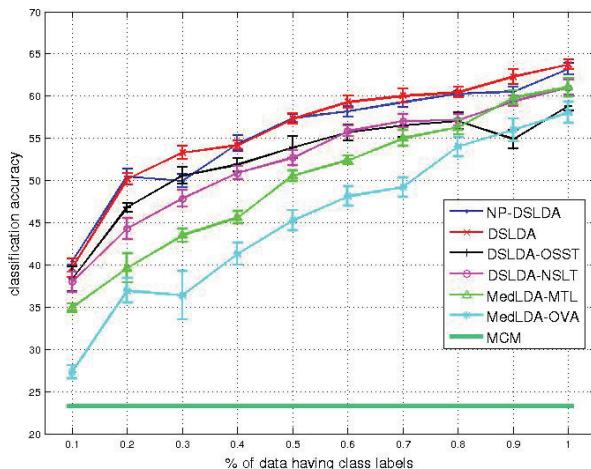
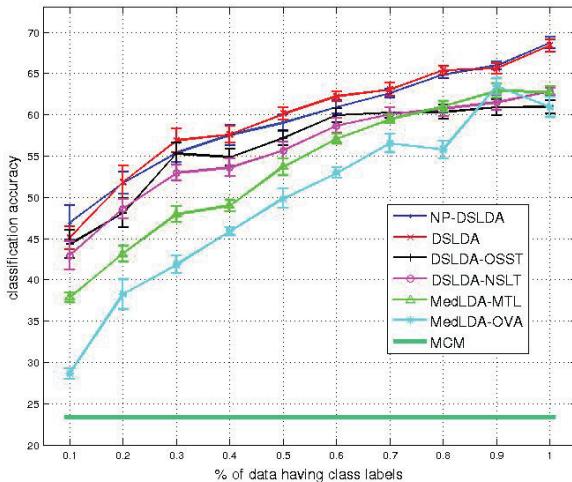


Fig. 3.23. $p_1 = 0.5$ (Conference).

MedLDA-OVA performs quite poorly when there is only a small amount of class supervision (note that this baseline uses *only* class labels). However, the per-

Fig. 3.24. $p_1 = 0.7$ (Conference).

formance approaches the others as the amount of class supervision increases. This is consistent with the intuition that multitask learning is most beneficial when each task has limited supervision and therefore has more to gain by sharing information with other tasks.

Shared supervised topics clearly increase classification accuracy when class supervision is limited (i.e. small values of p_2), as shown by the performance of both DSLDA-NSLT and DSLDA-OSST. When $p_2 = 1$ (equal amounts of topic and class supervision), DSLDA-OSST, MedLDA-MTL and MedLDA-OVA all perform similarly; however, by exploiting *both* types of supervision, DSLDA and NP-DSLDA still maintain a performance advantage.

3.5.3.1. Topic illustration

In Table 3.2, we show the most indicative words for several topics discovered by DSLDA from the text data (with $p_1 = 0.8$ and $p_2 = 1$). LT1 and LT2 correspond to the most frequent latent topics assigned to documents in the two broad categories of conferences (data mining and VLSI, respectively). The other five topics are supervised ones. CAD and IR stand for Computer Aided Design and Information Retrieval respectively. The illustrated topics are particularly discriminative when classifying documents.

Table 3.2. Illustration of Latent and Supervised Topics.

LT1	function, label, graph, classification, database, propagation, algorithm, accuracy, minimization, transduction
LT2	performance, design, processor, layer, technology, device, bandwidth, architecture, stack, system
CAD	design, optimization, mapping, pin, simulation, cache, programming, routing, biochip, electrode
VLSI	design, physical, lithography, optimization, interdependence, global, robust, cells, layout, growth
IR	algorithm, web, linear, query, precision, document, repair, site, search, semantics
Ranking	integration, catalog, hierarchical, dragpushing, structure, source, sequence, alignment, transfer, flattened, speedup
Learning	model, information, trajectory, bandit, mixture, autonomous, hierarchical, feedback, supervised, task

3.5.4. Methodology for experiments with Active Multitask Learning

Act-DSLDA and Act-NPDSLDA are compared against the following simplified models:

- Active Learning in MedLDA with **one-vs-all** classification (Act-MedLDA-OVA) (shown in Fig. 3.25): A separate MedLDA model is trained for each class using a one-vs-all approach leaving no possibility of transfer across classes. Supervised topics are not included in such modeling and the class labels are also obtained using active learning.
- Active Learning in MedLDA with **multitask** learning (Act-MedLDA-MTL) (shown in Fig. 3.26): A single MedLDA model is learned for all classes where the latent topics are shared across classes. Again, supervised topics are not used and the class labels are obtained using active learning. This baseline is intended to be stronger than Act-MedLDA-OVA where the latent topics are not shared.
- Act-DSLDA with **only shared supervised topics** (Act-DSLDA-OSST) (shown in Fig. 3.27): A model in which supervised topics are used and shared across classes but there are no latent topics. Both the supervised topics and the class labels are queried using active selection strategy.
- Act-DSLDA with **no shared latent topics** (Act-DSLDA-NSLT) (shown in Fig. 3.28): A model in which only supervised topics are shared across classes and a separate set of latent topics is maintained for each class. Both the supervised topics and the class labels are queried using active selection strategy. This model has richer representational capacity compared to Act-DSLDA-OSST which does not use any latent topics at all.
- Random selection of **only class labels** (MedLDA-MTL-Random) (shown in Fig. 3.29): A MedLDA-MTL model where examples with only class labels are selected at random but supervised topics are not used at all. Note that designing a DSLDA based model where only class labels are selected at random is tricky as one needs to balance the number of supervised topics queried and the number of class labels selected at random. This baseline shows the utility of active selection of classes in the MedLDA-MTL framework.

- Random selection of class and attribute labels (DSLDA-Random) (shown in Fig. 3.30): A DSLDA model where both queries for class and the supervised topics are selected at random. This baseline is weaker than RSC since the supervised topics are generally less informative compared to class labels. Both MedLDA-MTL-Random and DSLDA-Random are used to exhibit the utility of active learning for both class and supervised topic selection.

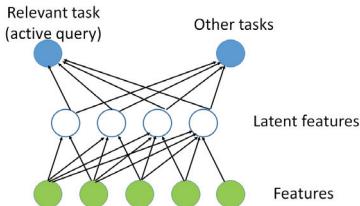


Fig. 3.25. Illustration of Act-MedLDA-OVA.

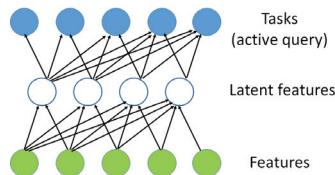


Fig. 3.26. Illustration of Act-MedLDA-MTL.

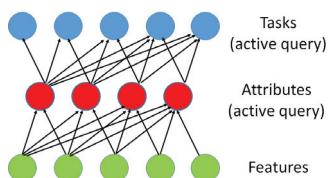


Fig. 3.27. Illustration of Act-DSLDA-OSST.

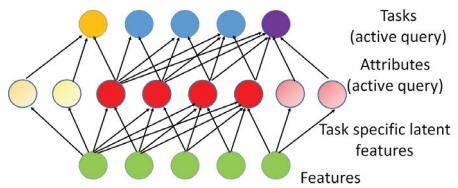


Fig. 3.28. Illustration of Act-DSLDA-NSLT.

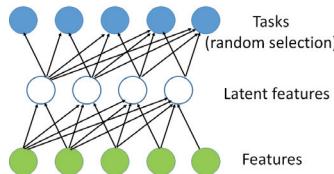


Fig. 3.29. Illustration of MedLDA-MTL-Random.

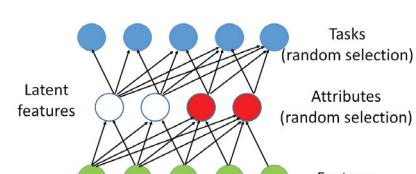


Fig. 3.30. Illustration of DSLDA-Random.

3.5.5. Active Multitask Learning results

For the experiments with active multitask learning, we start with a completely labeled dataset \mathcal{L} consisting of 300 documents. In every active iteration, we query for 50 labels (class labels or supervised topics). Figures 3.31 and 3.32 present representative learning curves for the image and the text data respectively, showing how classification accuracy improves as the amount of supervision is increased. The error bars in the curves show standard deviations across 20 trials.

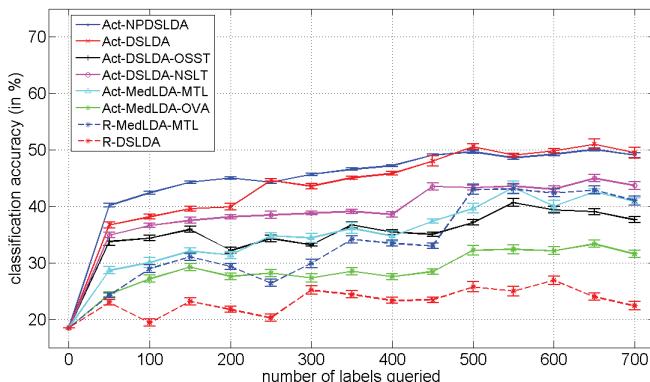


Fig. 3.31. aYahoo Learning Curves.

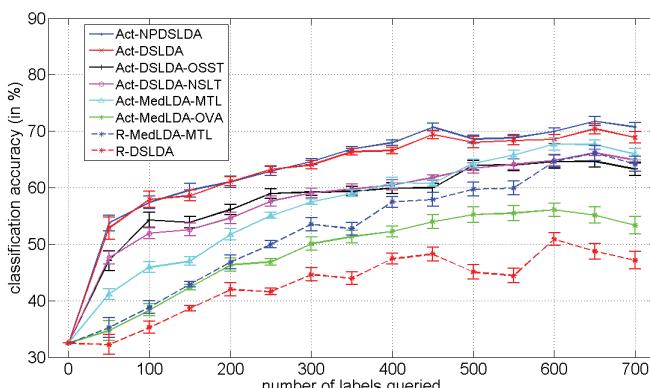


Fig. 3.32. ACM Conference Learning Curves.

3.5.6. Discussion

DSLDA-NSLT only allows sharing of supervised topics and its implementation is not straightforward. Since MedLDA-OVA, MedLDA-MTL and DSLDA use K topics (latent or a combination of supervised and latent), to make the comparison fair, it is necessary to maintain the same number of topics for DSLDA-NSLT. This ensures that the models compared have the same representational capacity. Therefore, for each class in DSLDA-NSLT, k_2/Y latent topics are maintained. While training DSLDA-NSLT with examples from the y^{th} class, only a subset of the first k_1 topics (or a subset of the supervised ones based on which of them are present in the training documents) and the next $(\frac{(y-1)k_2}{Y} + 1)^{\text{th}}$ to $(\frac{yk_2}{Y})^{\text{th}}$ topics are considered to be “active” among the latent topics. The other latent topics are assumed to have zero contribution, implying that the parameters associated with these topics are not updated based on observations of documents belonging to class y . During testing, however, one needs to project a document onto the entire K -dimensional space, and the class label is predicted based on this representation and the parameters r .

Overall, the results support the hypothesis that DSLDA’s ability to incorporate both supervised and latent topics allow it to achieve better predictive performance compared to baselines that exploit only one, the other, or neither. Furthermore, NP-DSLDA is able to automate model-selection, performing nearly as well as DSLDA with optimally chosen parameters.

Act-DSLDA and Act-NPDSLDA quite consistently outperform all of the baselines, clearly demonstrating the advantage of combining both types of topics and integrating active learning and transfer learning in the same framework. Act-NPDSLDA performs about as well or better as Act-DSLDA, for which the optimal number of latent topics has been chosen using an expensive model-selection search.

As to be expected, the active DSLDA methods’ advantage over their random selection counterpart (RSC) is greatest at the lower end of the learning curve. Act-MedLDA-OVA does a little better than RSCA showing that the active selection of class labels helps even if there is no transfer across classes. Act-MedLDA-MTL consistently outperforms Act-MedLDA-OVA as well as RSC showing that active transfer learning is beneficial for MedLDA-MTL. Act-DSLDA-OSST does better than both Act-MedLDA-MTL and RSC towards the lower end of the learning curve but with more labeled information this model does not perform that well since it does not use latent topics. Act-DSLDA-NSLT also performs better than Act-DSLDA-OSST because the former utilizes latent topics.

Figures 3.33 and 3.34 show the percentage (out of 50 queries) of class labels and supervised topics queried by Act-DSLDA at each iteration step in the vision and text data, respectively. Initially, the model queries for more class labels but towards the end of the learning curve, more supervised topics are queried. By the 14th iteration, the class labels of all the documents in the training set are queried. From the 15th iteration onwards, only supervised topics are queried. This observation is not that surprising since the class labels are more discriminative compared to the supervised topics and hence are queried more. However, queries of supervised topics are also helpful and allow continued improvement later in the learning curve.

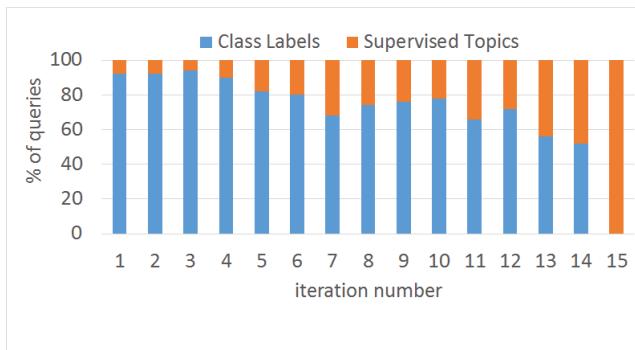


Fig. 3.33. aYahoo Query Distribution.

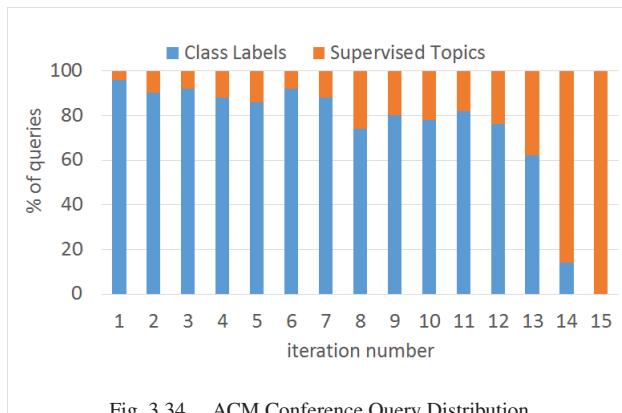


Fig. 3.34. ACM Conference Query Distribution.

3.6. Future Work and Conclusion

This chapter has introduced approaches that combine the following – generative and discriminative models, latent and supervised topics, and class and topic level supervision, in a principled probabilistic manner. Different ablations of the proposed models are also evaluated in order to understand the individual effects of latent/supervised topics, active learning and multitask learning on the overall model performance. The general idea of “double supervision” could be applied to many other models, for example, in multi-layer perceptrons, latent SVMs [54] or in deep belief networks [55]. In MTL, sharing tasks blindly is not always a good approach and further extension with clustered MTL [56] is possible. Further, sampling based algorithm could also be developed for solving the inference, possibly leading to even better performance without any factorized approximations.

Acknowledgments

This research was partially supported by ONR ATL Grant N00014-11-1-0105, NSF Grants (IIS-0713142 and IIS-1016614) and by the Brazilian Research Agencies FAPESP and CNPq.

References

- [1] I. Biederman, Recognition-by-components: A theory of human image understanding, *Psychological Review*. **94**, 115–147 (1987).
- [2] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. In *Proc. of CVPR*, pp. 1778–1785 (2009).
- [3] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by betweenclass attribute transfer. In *Proc. of CVPR*, pp. 951–958 (2009).
- [4] A. Kovashka, S. Vijayanarasimhan, and K. Grauman. Actively selecting annotations among objects and attributes. In *Proc. of ICCV*, pp. 1403–1410 (2011).
- [5] S. J. Pan and Q. Yang, A survey on transfer learning, *IEEE Transactions on Knowledge and Data Engineering*. **22**, 1345–1359 (2010).
- [6] R. Caruana, Multitask learning, *Machine Learning*. **28**, 41–75 (July, 1997).
- [7] D. M. Blei, A. Y. Ng, and M. I. Jordan, Latent Dirichlet Allocation, *Journal of Machine Learning Research*. **3**, 993–1022 (2003).
- [8] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning. Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora. In *Proc. of EMNLP*, pp. 248–256 (2009).
- [9] T. N. Rubin, A. Chambers, P. Smyth, and M. Steyvers. Statistical topic models for multi-label document classification. CoRR, abs/1107.2462 (2011).
- [10] D. M. Blei and J. D. McAuliffe. Supervised topic models. In *Proc. of NIPS* (2007).
- [11] J. Zhu, A. Ahmed, and E. P. Xing, MedLDA: maximum margin supervised topic

- models for regression and classification. In *Proc. Int. Conf. Machine Learning*, pp. 1257–1264 (2009).
- [12] J. Chang and D. Blei. Relational topic models for document networks. In *Proc. of AISTATS* (2009).
 - [13] K. D. Bollacker and J. Ghosh. Knowledge transfer mechanisms for characterizing image datasets. In *Soft Computing and Image Processing*. Physica-Verlag, Heidelberg (2000).
 - [14] S. Bickel, J. Bogojeska, T. Lengauer, and T. Scheffer. Multi-task learning for HIV therapy screening. In *Proceedings of International Conference on Machine Learning*, pp. 56–63, ACM, New York, NY, USA (2008). ISBN 978-1-60558-205-4.
 - [15] T. Evgeniou, M. Pontil, and O. Toubia, A convex optimization approach to modeling consumer heterogeneity in conjoint estimation, *Marketing Science*. **26**(6), 805–818 (Nov., 2007).
 - [16] R. K. Ando. Applying alternating structure optimization to word sense disambiguation. In *Proceedings of Computational Natural Language Learning* (2006).
 - [17] A. Torralba, K. P. Murphy, and W. T. Freeman, Sharing visual features for multiclass and multiview object detection, *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(5), 854–869 (May, 2007).
 - [18] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *Proc. Int. Conf. Machine Learning*, pp. 1113–1120 (2009).
 - [19] A. Passos, P. Rai, J. Wainer, and H. D. III. Flexible modeling of latent task structures in multitask learning. In *Proc. Int. Conf. Machine Learning*, pp. 1103–1110 (2012).
 - [20] A. Jalali, P. Ravikumar, S. Sanghavi, and C. Ruan. A dirty model for multi-task learning. In *Proc. of NIPS*, pp. 964–972 (2010).
 - [21] S. Kim and E. P. Xing. Tree-guided group lasso for multi-task regression with structured sparsity. In *Proc. Int. Conf. Machine Learning*, pp. 543–550 (2010).
 - [22] R. Jenatton, J. Audibert, and F. Bach, Structured variable selection with sparsity-inducing norms, *Journal of Machine Learning Research*. **12**, 2777–2824 (nov, 2011).
 - [23] J. Zhang, Z. Ghahramani, and Y. Yang, Flexible latent variable models for multi-task learning, *Machine Learning*. **73**(3), 221–242 (Dec., 2008).
 - [24] Y. Low, D. Agarwal, and A. J. Smola. Multiple domain user personalization. In *Proc. of KDD*, pp. 123–131 (2011).
 - [25] A. Quattoni, S. Wang, L. P. Morency, M. Collins, and T. Darrell. Hidden-state conditional random fields. In *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2007).
 - [26] T. Evgeniou, C. A. Micchelli, and M. Pontil, Learning multiple tasks with kernel methods, *Journal of Machine Learning Research*. **6**, 615–637 (2005). ISSN 1532-4435.
 - [27] S. Ben-David and R. Schuller. Exploiting task relatedness for multiple task learning. In *Learning Theory and Kernel Machines*, vol. 2777, pp. 567–580 (2003).
 - [28] A. Argyriou, C. A. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. In *Proc. of NIPS* (2007).
 - [29] R. Ando and T. Zhang, A framework for learning predictive structures from multiple tasks and unlabeled data, *Journal of Machine Learning Research*. **6**, 1817–1853 (2005). ISSN 1532-4435.

- [30] B. Bakker and T. Heskes, Task clustering and gating for Bayesian multitask learning, *Journal of Machine Learning Research.* **4** (2003).
- [31] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram, Multi-task learning for classification with Dirichlet process priors, *Journal of Machine Learning Research.* **8**, 35–63 (2007). ISSN 1532-4435.
- [32] L. Jacob, F. Bach, and J.-P. Vert, Clustered multi-task learning: A convex formulation, *CoRR. abs/0809.2085* (2008).
- [33] N. Roy and A. K. McCallum, Toward optimal active learning through sampling estimation of error reduction. In *Proc. Int. Conf. Machine Learning*, pp. 441–448 (2001).
- [34] B. Settles, Active learning literature survey, Computer Sciences Technical Report 1648, University of Wisconsin–Madison (2009).
- [35] G. Jun and J. Ghosh, An efficient active learning algorithm with knowledge transfer for hyperspectral data analysis. In *Geoscience and Remote Sensing Symposium, 2008. IGARSS 2008. IEEE International*, vol. 1, pp. I-52–I-55 (2008).
- [36] P. Rai, A. Saha, H. Daumé, III, and S. Venkatasubramanian, Domain adaptation meets active learning. In *Proc. of NAACL HLT Workshop on Active Learning for Natural Language Processing*, pp. 27–32 (2010).
- [37] Y. S. Chan and H. T. Ng, Domain adaptation with active learning for word sense disambiguation. In *Proc. of ACL*, pp. 49–56 (2007).
- [38] X. Shi, W. Fan, and J. Ren, Actively transfer domain knowledge. In *Proc. of ECML PKDD - Part II*, pp. 342–357 (2008).
- [39] A. Harpale and Y. Yang, Active learning for multi-task adaptive filtering. In *Proc. Int. Conf. Machine Learning*, pp. 431–438, Omnipress (2010).
- [40] S. Robertson and I. Soboroff, The TREC 2002 filtering track report. In *Text Retrieval Conference* (2002).
- [41] A. Saha, P. Rai, H. D. III, and S. Venkatasubramanian, Online learning of multiple tasks and their relationships., *JMLR - Proceedings Track.* **15**, 643–651 (2011).
- [42] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, Worst-case analysis of selective sampling for linear classification, *Journal of Machine Learning Research.* **7**, 1205–1230 (2006).
- [43] A. P. Dempster, N. M. Laird, and D. B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, *J. Royal Statistical Society. Series B (Methodological).* **39** (1), 1–38 (1977).
- [44] R. M. Neal and G. E. Hinton, A view of the EM algorithm that justifies incremental, sparse, and other variants. pp. 355–368, MIT Press, Cambridge, MA, USA (1999).
- [45] A. Bordes, L. Bottou, P. Gallinari, and J. Weston, Solving multiclass support vector machines with larank. In *Proc. Int. Conf. Machine Learning*, pp. 89–96 (2007).
- [46] A. Bordes, S. Ertekin, J. Weston, and L. Bottou, Fast kernel classifiers with online and active learning, *Journal of Machine Learning Research.* **6**, 1579–1619 (Dec., 2005).
- [47] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, LIBLINEAR: A library for large linear classification, *Journal of Machine Learning Research.* **9**, 1871–1874 (June, 2008).
- [48] Y. W. Teh, M. I. Jordan, M. J. Beal, and D. M. Blei, Hierarchical Dirichlet Processes, *Journal of the American Statistical Association.* **101**, 1566–1581 (December, 2006).
- [49] C. Wang, J. W. Paisley, and D. M. Blei, Online variational inference for the hierar-

- chical Dirichlet process, *JMLR - Proceedings Track*. **15**, 752–760 (2011).
- [50] J. Eisenstein, A. Ahmed, and E. P. Xing. Sparse additive generative models of text. In *Proceedings of International Conference on Machine Learning*, pp. 1041–1048 (2011).
 - [51] A. Acharya, A. Rawal, R. J. Mooney, and E. R. Hruschka. Using both supervised and latent shared topics for multitask learning. In *Proc. European Conference on Machine Learning and Principles and Practice of Knowledge Discovery, Part II, LNAI 8189*, pp. 369–384 (2013).
 - [52] D. G. Lowe, Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision*. **60**(2), 91–110 (Nov., 2004).
 - [53] C. Wang, B. Thiesson, C. Meek, and D. Blei. Markov topic models. In *Proceedings of Artificial Intelligence and Statistics* (2009).
 - [54] C. J. Yu and T. Joachims. Learning structural SVMs with latent variables. In *Proceedings of International Conference on Machine Learning*, pp. 1169–1176 (2009).
 - [55] G. E. Hinton and S. Osindero, A fast learning algorithm for deep belief nets, *Neural Computation*. **18**, 2006 (2006).
 - [56] J. Zhou, J. Chen, and J. Ye. Clustered Multi-Task Learning Via Alternating Structure Optimization. In *Proceedings of Neural Information Processing Systems* (2011).

Chapter 4

Sparse and Low-Rank Models for Visual Domain Adaptation

Rama Chellappa and Vishal M. Patel

*Center for Automation Research, UMIACS
University of Maryland, College Park, MD, USA*

rama@umiacs.umd.edu, pvishalm@umd.edu

In pattern recognition and computer vision, one is often faced with scenarios where the training data used to learn a model has different distribution from the data on which the model is applied. Regardless of the cause, any distributional change that occurs after learning a classifier can degrade its performance at test time. Domain adaptation tries to mitigate this degradation. This chapter presents an overview of recent domain adaptation methods based on sparse and low-rank representations.

4.1. Introduction

Supervised learning techniques have made tremendous contributions to machine learning and computer vision, leading to the development of robust algorithms that are applicable in practical scenarios. While these algorithms have significantly advanced the state-of-the-art, their performance is often limited by the amount of labeled training data available. Labeling is expensive and time consuming due to the significant amount of human efforts involved. However, collecting unlabeled visual data is becoming considerably easier due to the availability of low cost consumer and surveillance cameras, and large internet databases such as Flickr and YouTube. These data often come from multiple sources and modalities. Thus, when designing a classification or retrieval algorithm using these heterogeneous data, one has to constantly deal with the changing distribution of these data samples. Examples of such cases include: recognizing objects under poor lighting conditions and poses while algorithms are trained on well-illuminated objects at frontal pose, detecting and segmenting an organ of

interest from MRI images when available algorithms are instead optimized for CT and X-Ray images, recognizing and detecting human faces on infrared images while algorithms are optimized for color images, etc.

This challenge is commonly referred to as *covariate shift* [1] or *data set bias* [2, 3]. Any distributional change or domain shift that occurs after training can degrade the performance at test time. For instance, in the case of face recognition, to achieve useful performance *in the wild*, face representation and recognition methods must learn to adapt to distributions specific to each application domain shown in Fig. 4.1 (a)–(d). Domain adaptation tackles this problem by leveraging domain shift characteristics from labeled data in a related domain when learning a classifier for unseen data.

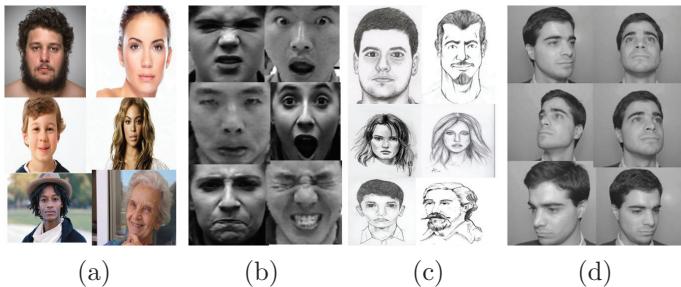


Fig. 4.1. (a) Unconstrained face images. (b) Images with expression variations. (c) Sketch images. (d) Images with pose variations. Real-world object recognition algorithms, such as face recognition, must learn to adapt to distributions specific to each domain shown in (a)–(d).

Although some special kinds of domain adaptation problems have been studied under different names such as covariate shift [1], class imbalance [4], and sample selection bias [5, 6], it only started gaining significant interest very recently in computer vision. There are also some closely related but not equivalent machine learning problems that have been studied extensively, including transfer learning or multi-task learning [7], self-taught learning [8], semi-supervised learning [9] and multiview analysis [10]. A review of domain adaptation methods from machine learning and natural language processing communities can be found in [11]. In this chapter, we review some recent sparse and low-rank representation-based domain adaptation approaches for computer vision applications [12, 13].

4.2. Sparse and Low-Rank Representations

In recent years, sparse and low-rank representation and dictionary learning have undergone rapid development, both in theory and in algorithms [14–16]. In this section, we briefly review sparse representation (also known as sparse coding), dictionary learning and sparse and low-rank decomposition.

4.2.1. Sparse Representation

Let \mathbf{D} be a redundant (overcomplete) dictionary with N atoms in \mathbb{R}^d

$$\mathbf{D} = [\mathbf{d}_1, \dots, \mathbf{d}_N] \in \mathbb{R}^{d \times N}.$$

The elements of \mathbf{D} are normalized to unit Euclidean norm i.e., $\|\mathbf{d}_i\| = 1 \quad \forall i$. Given a signal $\mathbf{y}_t \in \mathbb{R}^d$, finding the sparsest representation of \mathbf{y}_t in \mathbf{D} entails solving the following optimization problem

$$\mathbf{x}_t = \arg \min_{\mathbf{x}} \|\mathbf{x}\|_0 \text{ subject to } \mathbf{y}_t = \mathbf{D}\mathbf{x}, \quad (4.1)$$

where $\|\mathbf{x}\|_0 := \#\{j : x_j \neq 0\}$, which is a count of the number of nonzero elements in \mathbf{x} . Problem (4.1) is NP-hard and cannot be solved in a polynomial time. Hence, approximate solutions are usually sought. For instance, Basis Pursuit [17] offers the solution via ℓ_1 -minimization as

$$\mathbf{x}_t = \arg \min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ subject to } \mathbf{y}_t = \mathbf{D}\mathbf{x}, \quad (4.2)$$

where $\|\cdot\|_p$ for $0 < p < \infty$ is the ℓ_p -norm defined as

$$\|\mathbf{x}\|_p = \left(\sum_{j=1}^d |x_j|^p \right)^{\frac{1}{p}}.$$

The sparsest recovery is possible provided that certain conditions are met [16]. One can adapt the above framework to noisy settings, where the measurements are contaminated with an error \mathbf{n} obeying $\|\mathbf{n}\|_2 < \epsilon$, that is

$$\mathbf{y}_t = \mathbf{D}\mathbf{x} + \mathbf{n} \quad \text{for} \quad \|\mathbf{n}\|_2 < \epsilon. \quad (4.3)$$

A stable solution can be obtained by solving the following optimization problem [16]

$$\mathbf{x}_t = \arg \min_{\mathbf{x}} \|\mathbf{x}\|_1 \text{ subject to } \|\mathbf{y}_t - \mathbf{D}\mathbf{x}\|_2 < \epsilon. \quad (4.4)$$

4.2.2. Dictionary Learning

Traditionally, the dictionary \mathbf{D} in (4.1), is predetermined; e.g., wavelets. It has been observed that learning a dictionary directly from the training data rather than using a predetermined dictionary usually leads to a more compact representation and hence can provide improved results in many practical image processing applications such as restoration and classification [15, 16, 18].

Several algorithms have been developed for the task of learning a dictionary. Two of the most well-known algorithms are the method of optimal directions (MOD) [19] and the KSVD algorithm [20]. Given a set of examples $\mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_n]$, the goal of the KSVD and MOD algorithms is to find a dictionary \mathbf{D} and a sparse matrix \mathbf{X} that minimize the following representation error

$$(\hat{\mathbf{D}}, \hat{\mathbf{X}}) = \arg \min_{\mathbf{D}, \mathbf{X}} \|\mathbf{Y} - \mathbf{DX}\|_F^2 \text{ s. t. } \|\mathbf{x}_i\|_0 \leq T_0 \quad \forall i, \quad (4.5)$$

where \mathbf{x}_i represent the columns of \mathbf{X} , $\|\mathbf{A}\|_F$ denotes the Frobenius norm of \mathbf{A} and T_0 denotes the sparsity level. Both MOD and KSVD are iterative methods and alternate between sparse-coding and dictionary update steps. First, a dictionary \mathbf{D} with ℓ_2 normalized columns is initialized. Then, the main iteration is composed of the following two stages:

- *Sparse coding:* In this step, \mathbf{D} is fixed and the following optimization problem is solved to compute the representation vector \mathbf{x}_i for each example \mathbf{y}_i

$$i = 1, \dots, n, \quad \min_{\gamma_i} \|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2 \text{ s. t. } \|\mathbf{x}_i\|_0 \leq T_0.$$

- *Dictionary update:* This is where both MOD and KSVD algorithms differ. The MOD algorithm updates all the atoms simultaneously by solving an optimization problem whose solution is given by $\mathbf{D} = \mathbf{Y}\mathbf{X}^\dagger$, where \mathbf{X}^\dagger denotes the Moore-Penrose pseudo-inverse. Even though the MOD algorithm is very effective and usually converges in a few iterations, it suffers from the high complexity of the matrix inversion step as discussed in [20].

In the case of KSVD, the dictionary update is performed atom-by-atom in a computationally efficient way rather than using a matrix inversion. It has been observed that the KSVD algorithm requires fewer iterations to converge than the MOD method.

Dictionaries can be trained for both reconstruction and classification applications. In the late nineties, Etemand and Chellappa proposed a linear discriminant analysis (LDA) based basis selection and feature extraction algorithm for classification using wavelet packets [21]. Recently, similar algorithms for simultaneous sparse signal representation and discrimination have also been proposed [22–26].

Kernel-based non-linear sparse coding and dictionary learning methods have also been proposed in the literature [27–29]. These methods essentially map the input data onto a high-dimensional feature space using a predetermined kernel function. Sparse codes and dictionaries are then trained on the feature space for better representation and discrimination. Additional techniques for discriminative and kernel-based dictionary learning may be found within these references.

4.2.3. Sparse and Low-Rank Decomposition

Suppose that the given data samples are arranged as the columns of a matrix $\mathbf{Y} \in \mathbb{R}^{d \times n}$. One can estimate the low-dimensional subspace by finding a low-rank matrix \mathbf{L} such that the discrepancy between \mathbf{L} and \mathbf{Y} is minimized

$$\min_{\mathbf{L}, \mathbf{S}} \|\mathbf{S}\|_F \quad \text{s.t.} \quad \text{rank}(\mathbf{L}) \leq r, \quad \mathbf{Y} = \mathbf{L} + \mathbf{S}, \quad (4.6)$$

where $r \ll \min(d, n)$ and it is assumed that the data are corrupted by i.i.d. Gaussian noise. This problem can be solved by computing the Singular Value Decomposition (SVD) of \mathbf{Y} . Then projecting the columns of \mathbf{Y} onto the subspace spanned by the r principle left singular vectors of \mathbf{Y} . However, if the noise magnitude is large or if the entries of \mathbf{L} are arbitrarily corrupted then the above formulation breaks down. Recently, it was shown in [30] that as long as the error matrix \mathbf{S} is sufficiently sparse, one can exactly recover the low-rank matrix \mathbf{L} from $\mathbf{Y} = \mathbf{L} + \mathbf{S}$ by solving the following convex optimization problem

$$\min_{\mathbf{L}, \mathbf{S}} \|\mathbf{L}\|_* + \lambda \|\mathbf{S}\|_1 \quad \text{s.t.} \quad \mathbf{Y} = \mathbf{L} + \mathbf{S}, \quad (4.7)$$

where $\|\cdot\|_*$ denotes the nuclear norm of a matrix (i.e. the sum of its singular values), $\|\cdot\|_1$ denotes the sum of the absolute values of matrix entries, and λ is a positive parameter. The optimization problem (4.7) is referred to as Robust PCA (RPCA) [30].

4.3. Sparse and Low-Rank Representation-Based Visual Domain Adaptation

In this section, we give an overview of a number of recent visual domain adaptation methods that make use of sparse and low-rank modeling techniques. First, we define the problem of domain adaptation.

4.3.1. Domain Adaptation Problem Formulation

Let $\mathcal{S} = \{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^{N_s}$, where $\mathbf{x}^s \in \mathbb{R}^N$ denote the labeled data from the source domain. Here, \mathbf{x}^s is referred to as an observation and y^s is the corresponding class label. Labeled data from the target domain is denoted by $\mathcal{T}_l = \{(\mathbf{x}_i^{tl}, y_i^{tl})\}_{i=1}^{N_{tl}}$ where $\mathbf{x}^{tl} \in \mathbb{R}^M$. Similarly, unlabeled data in the target domain is denoted by $\mathcal{T}_u = \{\mathbf{x}_i^{tu}\}_{i=1}^{N_{tu}}$ where $\mathbf{x}^{tu} \in \mathbb{R}^M$. Unless specified otherwise, we assume $N = M$. Let $\mathcal{T} = \mathcal{T}_l \cup \mathcal{T}_u$. As a result, the total number of samples in the target domain is denoted by N_t which is equal to $N_{tl} + N_{tu}$. Denote $\mathbf{S} = [\mathbf{x}_1^s, \dots, \mathbf{x}_{N_s}^s]$ as the matrix of N_s data points from \mathcal{S} . Denote $\mathbf{T}_l = [\mathbf{x}_1^{tl}, \dots, \mathbf{x}_{N_{tl}}^{tl}]$ as the matrix of N_{tl} data from \mathcal{T}_l , $\mathbf{T}_u = [\mathbf{x}_1^{tu}, \dots, \mathbf{x}_{N_{tu}}^{tu}]$ as the matrix of N_{tu} data from \mathcal{T}_u and $\mathbf{T} = [\mathbf{T}_l | \mathbf{T}_u] = [\mathbf{x}_1^t, \dots, \mathbf{x}_{N_t}^t]$ as the matrix of N_t data from \mathcal{T} .

It is assumed that both the target and source data pertain to C classes or categories. Furthermore, it is assumed that all categories have some labeled data. We assume there is always a relatively large amount of labeled data in the source domain and a small amount of labeled data in the target domain. As a result, $N_s \gg N_{tl}$.

The goal of domain adaptation is to learn a function $f(\cdot)$ that predicts the class label of a novel test sample from the target domain. Depending on the availability of the source and target domain data, the domain adaptation problem can be defined in many different ways:

- In *semi-supervised domain adaptation*, the function $f(\cdot)$ is learned using the knowledge in \mathcal{S} and \mathcal{T}_l .
- In *unsupervised domain adaptation*, the function $f(\cdot)$ is learned using the knowledge in \mathcal{S} and \mathcal{T}_u .
- In *multi-source domain adaptation*, $f(\cdot)$ is learned from more than one domain in \mathcal{S} accompanying each of the above two cases.
- Finally, in the *heterogeneous domain adaptation*, the dimensions of features in the source and target domains are assumed to be different. In other words, $N \neq M$.

4.3.2. Domain Adaptive Dictionary Learning

When designing dictionaries, training and testing domains may be different, e.g., different view points and illumination conditions. In [31], a function learning framework is presented for the task of transforming a dictionary learned from one visual domain to the other, while maintaining a domain-invariant sparse representation of a signal. An overview of this method is shown in Fig. 4.2.

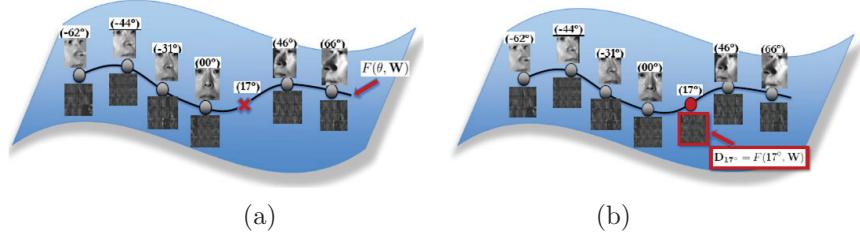


Fig. 4.2. Overview of the domain adaptive dictionary learning approach proposed in [31]. Consider example dictionaries corresponding to faces at different azimuths. (a) shows a depiction of example dictionaries over a curve on a dictionary manifold. Given example dictionaries, the approach presented in [31] learns the underlying dictionary function $F(\theta, \mathbf{W})$. In (b), the dictionary corresponding to a domain associated with observations is obtained by evaluating the learned dictionary function at corresponding domain parameters.

Denote P signals observed in N different domains as $\{\mathbf{Y}_1, \dots, \mathbf{Y}_N\}$, where $\mathbf{Y}_i = [\mathbf{y}_{i1}, \dots, \mathbf{y}_{iP}]$, $\mathbf{y}_{ip} \in \mathbb{R}^n$. Thus, \mathbf{y}_{ip} denotes the p^{th} signal observed in the i^{th} domain. Let \mathbf{D}_i denote the dictionary for the i^{th} domain, where $\mathbf{D}_i = [\mathbf{d}_{i1} \dots \mathbf{d}_{iK}]$, $\mathbf{d}_{ik} \in \mathbb{R}^n$. The domain dictionary learning problem can be formulated as

$$\arg \min_{\{\mathbf{D}_i\}_i^N, \mathbf{X}} \sum_i^N \|\mathbf{Y}_i - \mathbf{D}_i \mathbf{X}\|_F^2 \quad s.t. \quad \forall p \quad \|\mathbf{x}_p\|_o \leq T, \quad (4.8)$$

where $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_P]$, $\mathbf{x}_p \in \mathbb{R}^K$, are the sparse codes and T is a sparsity constant. The set of domain dictionary $\{\mathbf{D}_i\}_i^N$ learned through (4.8) enable the same sparse codes \mathbf{x}_p for a signal \mathbf{y}_p observed across N different domains to achieve domain adaptation.

A parametric function is used to model domain dictionaries \mathbf{D}_i as follows

$$\mathbf{D}_i = F(\theta_i, \mathbf{W}), \quad (4.9)$$

where $\boldsymbol{\theta}_i$ denotes a vector of domain parameters, e.g., view point angles, illumination conditions, etc., and \mathbf{W} denotes the dictionary function parameters [31]. Applying (4.9) to (4.8), one can formulate the domain dictionary function learning as follows

$$\arg \min_{\mathbf{W}, \mathbf{X}} \sum_i^N \|\mathbf{Y}_i - F(\boldsymbol{\theta}_i, \mathbf{W})\mathbf{X}\|_F^2 \quad s.t. \quad \forall p \quad \|\mathbf{x}_p\|_o \leq T. \quad (4.10)$$

Various linear and non-linear dictionary function learning models are considered in [31] and the optimization problem is solved using a simple three step procedure. See [31] for more details on the optimization of (4.10) and experimental results on various datasets.

4.3.3. Generalized Domain Adaptive Dictionary Learning

In [32] Shekhar *et al.* proposed a semi-supervised domain adaptive dictionary learning framework for learning a single dictionary to optimally represent both source and target data. As the features may not be correlated well in the original space, they propose to project data from both the domains onto a common low-dimensional space while maintaining the manifold structure of the data. They show that by learning the dictionary in a low-dimensional space, the algorithm can be made faster and irrelevant information in the original features can be discarded. Moreover, joint learning of dictionary and projections ensures that the common internal structure of data in both the domains is extracted, which can be represented well by sparse linear combinations of dictionary atoms. Figure 4.3 shown an overview of this method [32].

Given the source and target domain data $\mathbf{S} \in \mathbb{R}^{N \times N_s}$ and $\mathbf{T}_l \in \mathbb{R}^{M \times N_{tl}}$, respectively, Shekhar *et al.* learn a shared K atom dictionary, $\mathbf{D} \in \mathbb{R}^{l \times K}$ and mappings $\mathbf{W}_1 \in \mathbb{R}^{l \times N}$ and $\mathbf{W}_2 \in \mathbb{R}^{l \times M}$ onto a common low-dimensional space, which will minimize the representation error in the projected space. Formally, the following cost is minimized

$$\begin{aligned} \mathcal{C}_1(\mathbf{D}, \mathbf{W}_1, \mathbf{W}_2, \mathbf{X}_1, \mathbf{X}_2) = & \|\mathbf{W}_1 \mathbf{S} - \mathbf{D} \mathbf{X}_1\|_F^2 + \\ & \|\mathbf{W}_2 \mathbf{T}_l - \mathbf{D} \mathbf{X}_2\|_F^2 \end{aligned}$$

subject to sparsity constraints on $\mathbf{X}_1 \in \mathbb{R}^{K \times N_s}$ and $\mathbf{X}_2 \in \mathbb{R}^{K \times N_{tl}}$. It is assumed that rows of the projection matrices, \mathbf{W}_1 and \mathbf{W}_2 are orthogonal and normalized to unit-norm. This prevents the solution from becoming degenerate, leads to an efficient scheme for optimization and makes the kernelization of the algorithm possible. Note that this method does not

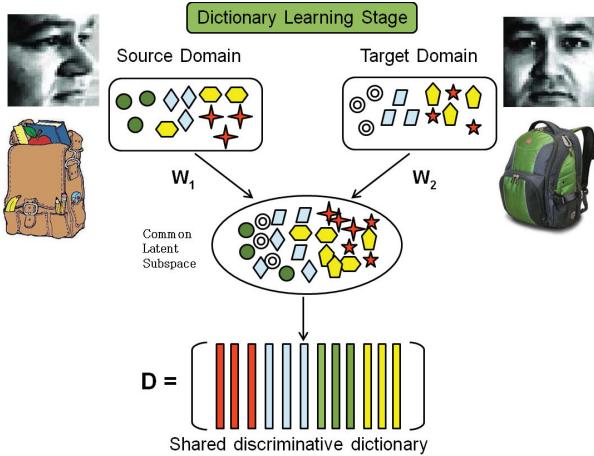


Fig. 4.3. Overview of the domain adaptive latent space dictionary learning framework [32].

require the data to be of same dimension in source and target domains. As a result, this method is applicable to heterogeneous domain adaptation problems [33].

In order to make sure that the projections do not lose too much information available in the original domains after projecting onto the latent space, a PCA-like regularization term is added which preserves energy in the original signal, given as

$$\begin{aligned} \mathcal{C}_2(\mathbf{W}_1, \mathbf{W}_2) = & \|\mathbf{S} - \mathbf{W}_1^T \mathbf{W}_1 \mathbf{S}\|_F^2 + \\ & \|\mathbf{T}_l - \mathbf{W}_2^T \mathbf{W}_2 \mathbf{T}_l\|_F^2. \end{aligned}$$

It is easy to show that the costs \mathcal{C}_1 and \mathcal{C}_2 , after ignoring the constant terms in \mathbf{Y} , can be written as

$$\mathcal{C}_1(\mathbf{D}, \tilde{\mathbf{W}}, \tilde{\mathbf{X}}) = \|\tilde{\mathbf{W}}\tilde{\mathbf{Y}} - \mathbf{D}\tilde{\mathbf{X}}\|_F^2, \quad (4.11)$$

$$\mathcal{C}_2(\tilde{\mathbf{W}}) = -\text{trace}((\tilde{\mathbf{W}}\tilde{\mathbf{Y}})(\tilde{\mathbf{W}}\tilde{\mathbf{Y}})^T) \quad (4.12)$$

where,

$$\tilde{\mathbf{W}} = [\mathbf{W}_1 \ \mathbf{W}_2], \ \tilde{\mathbf{Y}} = \begin{pmatrix} \mathbf{S} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_l \end{pmatrix}, \text{ and } \tilde{\mathbf{X}} = [\mathbf{X}_1 \ \mathbf{X}_2].$$

Hence, the overall optimization is given as

$$\begin{aligned} \{\mathbf{D}^*, \tilde{\mathbf{W}}^*, \tilde{\mathbf{X}}^*\} = & \underset{\mathbf{D}, \tilde{\mathbf{W}}, \tilde{\mathbf{X}}}{\text{argmin}} \mathcal{C}_1(\mathbf{D}, \tilde{\mathbf{W}}, \tilde{\mathbf{X}}) + \lambda \mathcal{C}_2(\tilde{\mathbf{W}}) \\ \text{s.t. } & \mathbf{W}_i \mathbf{W}_i^T = \mathbf{I}, \ i = 1, 2 \text{ and } \|\tilde{\mathbf{x}}_j\|_0 \leq T_0, \forall j \end{aligned} \quad (4.13)$$

where, λ is a positive constant. An efficient two-step procedure is proposed for solving this optimization problem in [32]. Furthermore, this method has been extended to multiple domains and kernelized in [32]. Once the projection matrices and the dictionary are learned, given a novel test sample from the target domain, it is first projected onto the latent domain using \mathbf{W}_2 and classified using a variation of the Latent Sparse Embedding Residual Classifier (LASERC) algorithm proposed in [34].

4.3.4. Subspace Interpolation via Dictionary Learning

Another dictionary-based domain adaptation method was recently proposed in [35]. Given labeled data in the source domain and unlabeled data in the target domain, the idea is to learn a set of intermediate domain and target domain to capture the intrinsic domain shift between two domains. Figure 4.4 shows an overview of this method.

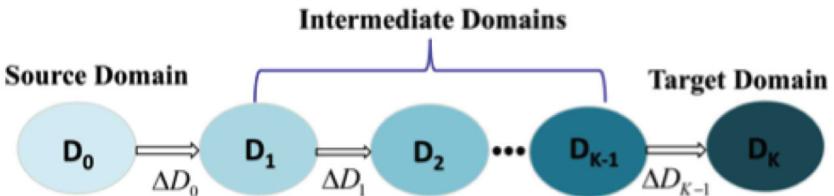


Fig. 4.4. An overview of the dictionary-based domain adaptation method [35].

Starting from the source domain dictionary \mathbf{D}_0 , the intermediate domain dictionaries $\{\mathbf{D}_k\}_{k=1}^K$ are sequentially learned to gradually adapt to the target data. The final dictionary \mathbf{D}_K which best represents the target data in terms of reconstruction error is taken as the target domain dictionary. Given the k th domain dictionary $\mathbf{D}_k, k \in [0, K - 1]$, the next domain dictionary \mathbf{D}_{k+1} is learned based on its coherence with \mathbf{D}_k and the remaining residue of the target data. Specifically, the target data \mathbf{T}_u are decomposed with \mathbf{D}_k and the residue \mathbf{J}_k are obtained by solving the following optimization problem

$$\mathbf{X}_k = \arg \min_{\mathbf{X}} \|\mathbf{T}_u - \mathbf{D}_k \mathbf{X}\|_F^2 \text{ s.t. } \forall \|\mathbf{x}_i\|_0 \leq T, \quad (4.14)$$

$$\mathbf{J}_k = \|\mathbf{T}_u - \mathbf{D}_k \mathbf{X}_k\|_F^2, \quad (4.15)$$

where \mathbf{X}_k are the sparse coefficients, \mathbf{x}_i is a column of \mathbf{X}_k , and T is the sparsity parameter. \mathbf{D}_{k+1} is then obtained by estimating $\Delta \mathbf{D}_k$ which is the

adjustment in the dictionary atoms between \mathbf{D}_{k+1} and \mathbf{D}_k

$$\min_{\Delta \mathbf{D}_k} \|\mathbf{J}_k - \Delta \mathbf{D}_k \mathbf{X}_k\|_F^2 + \lambda \|\Delta \mathbf{D}_k\|_F^2, \quad (4.16)$$

where λ is a parameter. An efficient optimization scheme is proposed for solving the above optimization problem. More details can be found in [35].

4.3.5. Hierarchical Sparse Representation-Based Domain Adaptation

Another recent work for visual domain adaptation using hierarchical networks was recently proposed by Nguyen *et al.* in [36]. Their method jointly learns a hierarchy of features together with transformations that address the mismatch between different domains. This method was motivated by [37] in which multi-layer sparse coding networks are proposed for building feature hierarchies layer by layer using sparse codes and spatial pooling. Figure 4.5 shows an overview of the sparse hierarchical domain adaptation

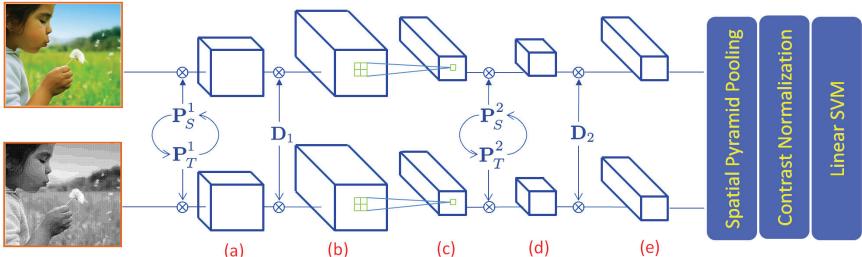


Fig. 4.5. An illustration of Domain Adaptation using a Sparse and Hierarchical Network (DASH-N) algorithm [36]. The source domain is RGB images and the target domain is halftone images. First, images are divided into small overlapping patches. These patches are vectorized while maintaining their spatial arrangements. (a) Performing contrast-normalization and dimensionality reduction using \mathbf{P}_S for source images and \mathbf{P}_T for target images. The circular feedbacks between \mathbf{P}_S and \mathbf{P}_T indicate that these two transformations are learned jointly. (b) Obtaining sparse codes using the common dictionary \mathbf{D}_1 . (c) Performing max pooling. The process then repeats for layer 2 (d & e), except that the input is the sparse codes from layer 1 instead of pixel intensities. At the final stage, spatial pyramid with max pooling are used to create image descriptors. Classification is done using linear support vector machine.

method [36]. The network contains multiple layers, each of which contains 3 sub-layers. The first sub-layer performs contrast-normalization and dimensionality reduction on the input data. Sparse coding is carried out in the second sub-layer. In the final sub-layer, adjacent features are max-pooled together to produce a new features. Output from one layer becomes

the input to the next layer. This method can be viewed as a generalization of the domain adaptive dictionary learning framework [32] using hierarchical networks. Extension of this method to multiple source domains has also been presented in [36].

4.3.6. Domain Adaptation with Low-Rank Reconstruction

Recently, a low-rank approximation based approach for semi-supervised domain adaptation was proposed in [38]. The basic goal of this method is to map the source data by a matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ to an intermediate representation where each transformed sample can be reconstructed by a linear combination of the target data samples

$$\mathbf{WS} = \mathbf{T}_l \mathbf{Z}, \quad (4.17)$$

where $\mathbf{Z} \in \mathbb{R}^{N_{tl} \times N_s}$ is the coefficient matrix. The following formulation is proposed to solve for the low-rank solution

$$\begin{aligned} (\hat{\mathbf{W}}, \hat{\mathbf{Z}}, \hat{\mathbf{E}}) &= \min_{\mathbf{W}, \mathbf{Z}, \mathbf{E}} \text{rank}(\mathbf{Z}) + \lambda \|\mathbf{E}\|_{2,1}, \\ \text{s.t. } \mathbf{WS} &= \mathbf{T}_l \mathbf{Z} + \mathbf{E}, \quad \mathbf{WW}^T = \mathbf{I}, \end{aligned} \quad (4.18)$$

where $\text{rank}(\cdot)$ denotes the rank of a matrix, λ is a parameter, $\mathbf{E} \in \mathbb{R}^{N \times N_s}$ is the error term and the $\ell_{2,1}$ -norm is defined as $\|\mathbf{E}\|_{2,1} = \sum_{i=1}^N \sqrt{\sum_{j=1}^{N_s} E_{ij}^2}$. Since the rank minimization problem (4.18) is NP-hard, the rank constraint is relaxed by the nuclear norm constraint [38]. The Augmented Lagrange Multiplier (ALM) method is proposed to solve the optimization problem.

Once the solution $(\hat{\mathbf{W}}, \hat{\mathbf{Z}}, \hat{\mathbf{E}})$ is obtained, the source data is transformed to the target domain as

$$\hat{\mathbf{WS}} - \hat{\mathbf{E}}. \quad (4.19)$$

The transformed source data are mixed with the target samples as the augmented training samples for training the classifiers. The trained classifier is then used to perform recognition on the unseen test samples in the target domain [38]. Extension of this method for the multiple source domain adaptation problem has also been proposed in [38].

4.4. Applications

In this section, we illustrate through different application examples the use and capabilities of various visual domain adaptation methods. In particular, we focus on object recognition and face recognition applications.

4.4.1. Face Recognition

Face recognition is a challenging problem that has been actively researched for over two decades [39]. Current systems work very well when training and test images are captured under controlled conditions. However, their performance degrades significantly when the test images contain variations that are not present in the training images. One of these variations is change in pose. Along with the frontal images with different illumination (source images), if we are also given a few images at different poses (target images), then the resulting face recognition problem can be viewed a domain adaptation problem [31, 32, 40].

Face recognition experiments were conducted on the CMU Multi-PIE dataset [41] with images of 129 subjects in frontal pose as the source domain, and five other off-frontal poses as the target domain. Images under five illumination conditions across source and target domains were used for training with which images from remaining 15 illumination conditions in the target domain were recognized. Results provided in Table 4.1 show that the dictionary-based adaptation method [32] compares favorably with some of the recently proposed multi-view recognition algorithms [10] as well as many other non-adaptation techniques, and gives the best performance on average. Note that the discriminative dictionary learning algorithm, FDDL [25] does not provide the best results here as it is not able to efficiently represent the non-linear changes introduced by the pose variation.

Table 4.1. Comparison of various algorithms for face recognition across pose [32].

Method	Probe pose					Average
	15°	30°	45°	60°	75°	
PCA	15.3	5.3	6.5	3.6	2.6	6.7
PLS [42]	39.3	40.5	41.6	41.1	38.7	40.2
LDA	98.0	94.2	91.7	84.9	79.0	89.5
CCA [42]	92.1	89.7	88.0	86.1	83.0	83.5
GMLDA [10]	99.7	99.2	98.6	94.9	95.4	97.6
FDDL [25]	96.8	90.6	94.4	91.4	90.5	92.7
SDDL [32]	98.4	98.2	98.9	99.1	98.8	98.7

Furthermore, the learned dictionaries were also used for pose alignment where the goal is to align faces from one pose to a different pose. This is a challenging problem since actual pose variations are three dimensional whereas the image evidence one has is two dimensional. Sample results are shown in Figure 4.6. One of the interesting features of the dictionary-based

adaptation methods is that they allow the synthesis of data associated with different domains while exploiting the generative power of dictionary-based representations. This is essentially what is highlighted in the last two rows of Figure 4.6. The dictionary-based method is robust at high levels of noise and missing pixels. It produces denoised and inpainted synthesized images. Additional results on various face recognition tasks using domain adaptation can be found in [31] and [35].

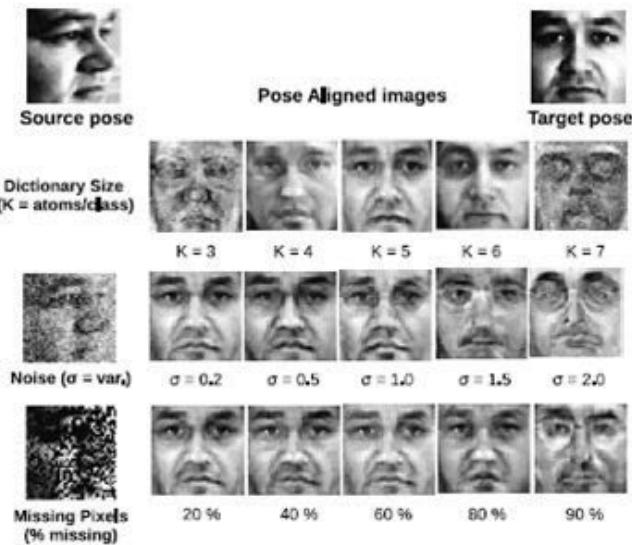


Fig. 4.6. Examples of pose-aligned images. Synthesis in various conditions demonstrate the robustness of the domain adaptive dictionary learning method [32].

4.4.2. Object Recognition

In this section, we compare the performance of various visual domain adaptation methods on a benchmark object recognition dataset which was introduced in [43]. The dataset consists of images from three sources: Amazon (consumer images from online merchant sites), DSLR (images by DSLR camera) and Webcam (low quality images from webcams). In addition, algorithms are tested on the Caltech-256 dataset [44], taking it as the fourth domain. Figure 4.7 shows sample images from these datasets, and clearly highlights the differences between them.



Fig. 4.7. Example images from KEYBOARD and BACK-PACK categories in Caltech-256, Amazon, Webcam and DSLR. Caltech-256 and Amazon datasets have diverse images, Webcam and DSLR are similar datasets with mostly images from offices [32].

Three set-ups are followed for comparing the performance of various algorithms. In the first set-up, 10 classes: BACKPACK, TOURING-BIKE, CALCULATOR, HEADPHONES, COMPUTER-KEYBOARD, LAPTOP-101, COMPUTER-MONITOR, COMPUTER-MOUSE, COFFEE-MUG, AND VIDEO-PROJECTOR, common to all the four datasets are used. In this case, there are a total of 2533 images. Each category has 8 to 151 images in a dataset. In the second set-up, all 31 classes from Amazon, Webcam and DSLR are used to evaluate various algorithms. Finally, in the third set-up, methods for adaptation are evaluated using multiple domains. In this case, the first dataset is used and methods are tested on all the 31 classes in it. For both the cases, we use 20 training samples per class for Amazon/Caltech, and 8 samples per class for DSLR/Webcam when used as source, and 3 training samples for all of them when used for target domain. Rest of the data in the target domain is used for testing. The experiment is run multiple times for random train/test splits and the

result is averaged over all the runs. For the unsupervised case, the same setting as semi-supervised adaptation described above is followed but without using any labeled data from the target domain^a.

4.4.2.1. Semi-supervised Adaptation Results using Single Source

The semi-supervised adaptation recognition results of different algorithms on 8 pairs of source-target domains and on all 31 classes are shown in Table 4.2 and Table 4.3, respectively. Baseline results obtained using the Hierarchical Matching Pursuit (HMP) method [37] as well as the Fisher Discrimination Dictionary Learning (FDDL) method [25] which learn the dictionaries separately for the source and target domains without performing domain adaptation are also included.

Table 4.2. Semi-supervised domain adaptation results of different approaches on four domains with 10 common classes (C: Caltech, A: Amazon, D: DSLR, W: Webcam).

Methods	C → A	C → D	A → C	A → W	W → C	W → A	D → A	D → W
Metric [43]	33.7 ± 0.8	35.0 ± 1.1	27.3 ± 0.7	36.0 ± 1.0	21.7 ± 0.5	32.3 ± 0.8	30.3 ± 0.8	55.6 ± 0.7
SGF [46]	40.2 ± 0.7	36.6 ± 0.8	37.7 ± 0.5	37.9 ± 0.7	29.2 ± 0.7	38.2 ± 0.6	39.2 ± 0.7	69.5 ± 0.9
GFK [47]	46.1 ± 0.6	55.0 ± 0.9	39.6 ± 0.4	56.9 ± 1.0	32.8 ± 0.1	46.2 ± 0.6	46.2 ± 0.6	80.2 ± 0.4
FDDL [25]	39.3 ± 2.9	55.0 ± 2.8	24.3 ± 2.2	50.4 ± 3.5	22.9 ± 2.6	41.1 ± 2.6	36.7 ± 2.5	65.9 ± 4.9
HMP [37]	67.7 ± 2.3	70.2 ± 5.1	51.7 ± 4.3	70.0 ± 4.2	46.8 ± 2.1	61.5 ± 3.8	64.7 ± 2.0	76.0 ± 4.0
SDDL [32]	49.5 ± 2.6	76.7 ± 3.9	27.4 ± 2.4	72.0 ± 4.8	29.7 ± 1.9	49.4 ± 2.1	48.9 ± 3.8	72.6 ± 2.1
DASH-N [36]	71.6 ± 2.2	81.4 ± 3.5	54.9 ± 1.8	75.5 ± 4.2	50.2 ± 3.3	70.4 ± 3.2	68.9 ± 2.9	77.1 ± 2.8

Table 4.3. Single-source semi-supervised domain adaptation results on all 31 classes.

Method	A → W	D → W	W → D
Metric [43]	44	31	27
RDALR [38]	50.7 ± 0.8	36.9 ± 19.9	32.9 ± 1.2
SGF [46]	57 ± 3.5	36 ± 1.1	37 ± 2.3
GFK [47]	46.4 ± 0.5	61.3 ± 0.4	66.3 ± 0.4
HMP [37]	55.7 ± 2.5	50.5 ± 2.7	56.8 ± 2.6
SDDL [32]	50.1 ± 2.5	51.2 ± 2.1	50.6 ± 2.6
DASH-N [36]	60.6 ± 3.5	67.9 ± 1.1	71.1 ± 1.7

Compared to the metric learning-based approach [43], manifold-based feature concatenation methods [46, 47] provide better results. This makes sense because by finding intermediate domain representations one is able to learn a feature vector that is more robust than a feature vector that results by learning a single transformation that minimizes the effect of the domain shift. The SDDL method can be viewed as an extension of the FDDL

^aSeveral recent methods explore both source and target data at once in a transductive manner rather than splitting the datasets into multiple training/testing partitions. See [45] for details on the evaluation protocol using this setting.

method which simultaneously learns discriminative dictionaries on a latent space where both the source and the target data are forced to have similar sparse representation. As a result, one can clearly see the performance gain of the SDDL method over the FDDL method as well as the manifold-based methods in Table 4.2 and Table 4.3.

The HMP method [37] builds a feature hierarchy layer by layer using an efficient matching pursuit encoder. It consists of three main components: batch tree orthogonal matching pursuit, spatial pyramid matching, and contrast normalization. As a results, it is robust to some of the variations present in the images such as illumination changes, pose variations and resolution variations. The DASH-N method essentially extends the SDDL and HMP methods by learning features directly from data for domain adaptation. As a result, it provides more robust and discriminative representation of the data and performs the best on this dataset on both settings. The dictionary learning-based methods [25, 37] essentially find the common internal structure of the data. They inherently have the denoising capability and provide robust representation of the data. This is one of the reasons why in some cases the FDDL and the HMP methods provide better results than metric learning and manifold-based methods.

4.5. Conclusions and Future Directions

In this chapter, we provided an overview of recent developments in domain adaptation for computer vision using sparse and low-rank models, with an emphasis on applications to the problems of face and object recognition. We believe the availability of massive data has brought substantial opportunities and challenges to the analysis of datasets bias or covariant shifts, and domain adaptation problems.

Domain adaptation promises to be an active area of research, especially as one of the possible ways to quickly propagate semantic annotations to the large-scale visual data being acquired at every minute. In computer vision, researchers have identified specific challenges that do not belong to machine learning: a major question among them that is rarely addressed in traditional domain adaptation research is one of adapting structured (non-vector) data representations. In machine learning or natural language processing, an input sample is usually represented as a vector in Euclidean space, different samples are treated as independent observations, and the task is typically classification. This is, however, not the case in computer vision where the representations to be potentially adapted include shapes

and contours, deformable and articulated 2-D or 3-D objects, graphs and random fields, intrinsic images, as well as visual dynamics, none of which is directly supported by “vectorial” domain adaptation techniques. In addition to recognition and detection, models and algorithms for segmentation, reconstruction, and tracking are awaiting mechanisms that do not yet exist, to be adapted toward emerging new domains. All of these challenges necessitate continuous efforts on characterizing visual domain shift and a paradigm of effective and efficient adaptation methods that are dedicated to visual data.

In the meantime, it is generally accepted that domain shifts in computer vision are usually due to causes from the imaging process that can be explained physically, such as illumination changes, sensor changes, view point changes, etc. We believe incorporating these physical priors into strong statistical adaptation approaches will not only lead to performance increase, but also lead to other insights in understanding the imaging process. This calls for a “physically-informed” adaptation paradigm that better exploits knowledge about image formation and better integrates other domain-specific knowledge implied by the diverse set of partial, noisy, and multimodal “side information” accompanying the visual data, such as imagery obtained from online social media. We hope that by appropriately incorporating physically-informed adaptation paradigm, distributional changes across different sensors (EO/SAR, IR/SAR, Eo/IR, etc.) can be handled.

Finally, we expect that studies on data characteristics and adaptations will produce stronger guidance to developing more desirable datasets for evaluating research in a wider spectrum of computer vision problems.

Acknowledgment

This work was supported by a MURI from the Office of Naval Research under the Grant 1141221258513 and a MURI from the Army Research Office under the Grant W911NF-09-1-0383.

References

- [1] H. Shimodaira, Improving predictive inference under covariate shift by weighting the log-likelihood function, *Journal of Statistical Planning and Inference*. **90**(2), 227–244 (2000).
- [2] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *IEEE Conference on Computer Vision and Pattern Recognition* (2011).

- [3] A. Khosla, T. Zhou, T. Malisiewicz, A. A. Efros, and A. Torralba. Undoing the damage of dataset bias. In *European Conference on Computer Vision* (2012).
- [4] N. Japkowicz and S. Stephen, The class imbalance problem: A systematic study, *Intelligent Data Analysis*. **6**(5), 429–450 (2002).
- [5] J. J. Heckman, Sample selection bias as a specification error, *Econometric*. **47**(1), 153–161 (1979).
- [6] B. Zadrozny. Learning and evaluating classifiers under sample selection bias. In *International Conference on Machine Learning*, pp. 114–121 (2004).
- [7] R. Caruana, Multitask learning, *Machine Learning*. **28**(1), 41–75 (1997).
- [8] R. Raina, A. Battle, H. Lee, B. Packer, and A. Y. Ng. Self-taught learning: transfer learning from unlabeled data. In *International conference on Machine learning*, pp. 759–766 (2007).
- [9] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised learning*. MIT Press (2006).
- [10] A. Sharma, A. Kumar, H. Daume, and D. Jacobs. Generalized multiview analysis: A discriminative latent space. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2160–2167 (2012).
- [11] J. Jiang. *Domain adaptation in natural language processing*. PhD thesis, University of Illinois at Urbana-Champaign (2008).
- [12] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa, Visual domain adaptation: A survey of recent advances, *IEEE Signal Processing Magazine* (2014).
- [13] V. M. Patel, Y.-C. Chen, R. Chellappa, and P. J. Phillips, Dictionaries for image and video-based face recognition, *Journal of the Optical Society of America A*. **31**(5), 1090–1103 (May, 2014).
- [14] J. Wright, Y. Ma, J. Mairal, G. Sapiro, T. S. Huang, and S. Yan, Sparse representation for computer vision and pattern recognition, *Proceedings of the IEEE*. **98**(6), 1031–1044 (2010).
- [15] R. Rubinstein, A. M. Bruckstein, and M. Elad, Dictionaries for sparse representation modeling, *Proceedings of the IEEE*. **98**(6), 1045 –1057 (june, 2010).
- [16] M. Elad, *Sparse and Redundant Representations: From theory to applications in Signal and Image processing*. Springer (2010).
- [17] S. Chen, D. Donoho, and M. Saunders, Atomic decomposition by basis pursuit, *SIAM J. Sci. Comp.* **20**(1), 33–61 (1998).
- [18] B. A. Olshausen and D. J. Fieldt, Sparse coding with an overcomplete basis set: a strategy employed by v1, *Vision Research*. **37**, 3311–3325 (1997).
- [19] K. Engan, S. O. Aase, and J. H. Husoy, Method of optimal directions for frame design, *IEEE International Conference on Acoustic, Speech, Signal Process.* **5**, 2443–2446 (1999).
- [20] M. Aharon, M. Elad, and A. M. Bruckstein, The k-svd: an algorithm for designing of overcomplete dictionaries for sparse representation, *IEEE Transaction on Signal Process.* **54**(11), 4311–4322 (2006).
- [21] K. Etemand and R. Chellappa, Separability-based multiscale basis selection and feature extraction for signal and image classification, *IEEE Transactions on Image Processing*. **7**(10), 1453–1465 (Oct. 1998).

- [22] J. Mairal, F. Bach, and J. Ponce, Task-driven dictionary learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **34**(4), 791–804 (2012).
- [23] Z. Jiang, Z. Lin, and L. S. Davis, Label consistent k-svd: Learning a discriminative dictionary for recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **35**(11), 2651–2664 (Nov, 2013).
- [24] Q. Zhang and B. Li. Discriminative k-svd for dictionary learning in face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition* (2010).
- [25] M. Yang, L. Zhang, X. Feng, and D. Zhang. Fisher discrimination dictionary learning for sparse representation. In *International Conference on Computer Vision*, pp. 543–550 (2011).
- [26] Q. Qiu, V. M. Patel, and R. Chellappa, Information-theoretic dictionary learning for image classification, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **36**(11), 2173–2184 (2014).
- [27] H. V. Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa, Design of non-linear kernel dictionaries for object recognition, *IEEE Transactions on Image Processing*. **22**(12), 5123–5135 (2013).
- [28] A. Shrivastava, H. V. Nguyen, V. M. Patel, and R. Chellappa. Design of non-linear discriminative dictionaries for image classification. In *Asian Conference on Computer Vision (ACCV)* (2012).
- [29] L. Zhang, W.-D. Zhou, P.-C. Chang, J. Liu, Z. Yan, T. Wang, and F.-Z. Li, Kernel sparse representation-based classifier, *IEEE Transactions on Signal Processing*. **60**(4), 1684 –1695 (april, 2012).
- [30] E. J. Cands, X. Li, Y. Ma, J. Wright, and R. Chellappa, Robust principal component analysis?, *Journal of ACM*. **58**(1), 1–37 (2009).
- [31] Q. Qiu, V. M. Patel, P. Turaga, and R. Chellappa. Domain adaptive dictionary learning. In *European Conference on Computer Vision*, vol. 7575, pp. 631–645 (2012).
- [32] S. Shekhar, V. M. Patel, H. V. Nguyen, and R. Chellappa. Generalized domain-adaptive dictionaries. In *IEEE Conference on Computer Vision and Pattern Recognition* (2013).
- [33] W. Li, L. Duan, D. Xu, and I. Tsang, Learning with augmented features for supervised and semi-supervised heterogeneous domain adaptation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **PP**(99), 1–1 (2013).
- [34] H. V. Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa. Sparse embedding: A framework for sparsity promoting dimensionality reduction. In *European conference on Computer vision* (2012).
- [35] J. Ni, Q. Qiu, and R. Chellappa. Subspace interpolation via dictionary learning for unsupervised domain adaptation. In *IEEE International Conference on Computer Vision* (2013).
- [36] H. V. Nguyen, H. T. Ho, V. M. Patel, and R. Chellappa, Joint hierarchical domain adaptation and feature learning, *IEEE Transactions on Image Processing* (2014).

- [37] L. Bo, X. Ren, and D. Fox. Hierarchical Matching Pursuit for Image Classification: Architecture and Fast Algorithms. In *Neural Information Processing Systems*, pp. 2115–2123 (2011).
- [38] I.-H. Jhuo, D. Liu, D. Lee, and S.-F. Chang. Robust visual domain adaptation with low-rank reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2168–2175 (2012).
- [39] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld, Face recognition: A literature survey, *ACM Computing Surveys*. pp. 399–458 (Dec. 2003).
- [40] H. Ho and R. Gopalan, Model-driven domain adaptation on product manifolds for unconstrained face recognition, *International Journal of Computer Vision* (2014).
- [41] R. Gross, I. Matthews, J. F. Cohn, T. Kanade, and S. Baker, Multi-pie, *Image Vision Computing*. **28**(5), 807–813 (2010).
- [42] A. Sharma and D. Jacobs. Bypassing synthesis: Pls for face recognition with pose, low-resolution and sketch. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593–600 (2011).
- [43] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision*, vol. 6314, pp. 213–226 (2010).
- [44] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology (2007). URL <http://authors.library.caltech.edu/7694>.
- [45] B. Gong, K. Grauman, and F. Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *International Conference on Machine Learning* (2013).
- [46] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *IEEE International Conference on Computer Vision*, pp. 999–1006 (2011).
- [47] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2066–2073 (2012).

Chapter 5

Pattern Classification using the Principle of Parsimony: Two Examples

Jayanta Basak

*NetApp India Private Limited
Bangalore, India*

basak@netapp.com, basakjayanta@yahoo.com

The Principle of Parsimony or the Occam's Razor is a key principle for improving the generalization performance in pattern classification. The principle is essentially equivalent to reducing the variance of a classifier at the cost of increased boundary bias. In this paper, we provide a quantitative way to express this principle in terms of the outcome of a classifier instead of explicitly regularizing the model complexity in terms of model parameters. We then use this principle to design a new kernel machine and a modified k -nearest neighbor algorithm. Experimentally we validate the performance of these two classifiers over real-life datasets. We also discuss the relationship of the proposed kernel machine with several other existing kernel machines.

5.1. Introduction

In pattern classification [1–3], the loss or discrepancy $L(t, f(x, \alpha))$ between the desired response t to a given input x and the response provided by a learning machine $f(x, \alpha)$ is minimized with respect to the distribution $F(x, t)$. The goal is to minimize the risk functional

$$R(\alpha) = \int L(t, f(x, \alpha)) dF(x, t) \quad (5.1)$$

and obtain an α_0 such that $f(x, \alpha_0)$ minimizes $R(\alpha)$. The general expression of the risk functional is quite broad to encompass various specific problems such as pattern classification, regression, and density estimation. In the present article, we restrict ourselves only to the problem of pattern classification.

In two-class classification tasks, usually the desired response t can take two different values e.g., $t \in \{-1, 1\}$ and $f(x, \alpha)$ represents a set of indicator functions (with two possible values of -1 and 1) for different values of α . The loss function is then defined as

$$L(t, f(x, \alpha)) = \begin{cases} 0 & \text{if } t = f(x, \alpha) \\ 1 & \text{if } t \neq f(x, \alpha) \end{cases} \quad (5.2)$$

Since the distribution $F(x, t)$ is unknown, the loss functional is often replaced by the empirical risk functional

$$R_{emp}(\alpha) = \frac{1}{N} \sum_{i=1}^N L(t_i, f(x_i, \alpha)) \quad (5.3)$$

where $L(t_i, f(x_i, \alpha))$ is the error or loss on the observed pair (x_i, t_i) . Assuming the Gaussian error model for the loss function [4], the maximum likelihood estimate of the empirical risk functional reduces to the least square estimate such that

$$R_{emp}(\alpha) = \frac{1}{N} \sum_{i=1}^N (t_i - f(x_i, \alpha))^2 \quad (5.4)$$

The empirical risk minimizes the error on the observed samples (training set) by finding out one optimal α depending on $f(\cdot)$. In general, minimization of the empirical risk functional with increased model complexity increases the generalization error i.e., the error on the unobserved data [2, 5–8].

Various principles are applied to improve the generalization performance which include [9–21]). In general, the basic spirit of all these approaches is to reduce the unnecessary model complexities while performing the classification by minimizing the empirical risk functional. This basic principle of reducing unnecessary complexity is often referred to as Occam's razor or the principle of parsimony [2]. The different forms for realizing this principle include Tikhonov regularization [1, 22], minimum description length principle [23], and structural risk minimization [3, 24]. For example, in Tikhonov regularization, normally an additional cost dependent on the model complexity is added to the empirical loss functional. Such a cost functional imposes a smoothing on the classification boundary. Minimum description length (MDL) principle is based on the information theoretic analysis of the randomness in the concept. As discussed in [2], according to MDL, the sum of the classifier's algorithmic complexity and the description of the training data with respect to the classifier is minimized. Different

variations of the MDL principle such as the Akaike information criterion (AIC) [25], the Bayes information criterion (BIC) [26] and the network information criterion (NIC) [27] are available in the literature. In structural risk minimization (SRM), on the other hand, the complexity of a classifier is measured in terms of the Vapnik-Chervonenkis (VC) dimension [3, 24]. SRM principle finds a trade-off between model complexity (VC confidence) and the empirical error of model fitting. According to this principle, functions or models in the hypotheses space are arranged into a hierarchy of nested subsets of increasing VC-dimension.

As articulated in [5], the overall classification error as measured by the deviation of a model's performance with respect to a theoretically optimal classifier (Bayes classifier), can be expressed as a nonlinear relationship between the boundary bias and the variance. The performance of a classifier is more dependent on the variance component and a reduced variance can enhance the classification performance. However, the reduction in variance can cause an increase in the boundary bias due to over-smoothing. The increase in boundary bias does not necessarily deteriorate the generalization performance of a classifier so long as the sign of boundary bias remains consistent. In the existing realizations of the principle of parsimony, essentially the variance is reduced by increasing the bias, and the bias is increased to the extent such that the sign remains consistent.

In this paper, we propose an alternate approach of reducing the variance component and provide a simple quantitative way of expressing the principle of parsimony for classification. According to this principle, we essentially reduce the superfluity in the outcome (variance) constrained by a certain regularization parameter. We first articulate the principle and then explain the principle in the perspective of the bias-variance decomposition. We then design a kernel-based classifier using this principle. We also use the same principle for designing a modified k-nearest neighbor classifier. For the kernel based classifier, we draw the relationships with other existing kernel machines. We show the effectiveness of this principle by demonstrating the classification performance of our kernel machine and the enhanced k-nearest neighbor classifier.

5.2. Pattern Classification with the Principle of Parsimony

5.2.1. Principle

We present an interpretation of the principle of parsimony which we use in the subsequent design of our classifier. We state the principle as:

Minimize the superfluity in the outcome with a given margin.

Here we use this principle to regularize the outcome of a classifier which in turn regularizes the model complexity. Using this principle we design classifiers without directly considering the minimization of model complexity in the loss functional. Rather we design classifiers such that the outcomes of the models follow this principle. At a later part of this paper, we show that the principle can be connected to the model complexity under certain constraints. Note that, this principle is not new; however, we express the principle of regularization in this form which helps us in designing two different forms of classifiers. We first explain the notion of the “superfluity” that we use in the sequel restricting ourselves to two-class classification tasks. We then draw the connection with the bias-variance perspective of the same principle. Subsequently, we design two different classifiers using the same principle.

Two Class Classification: In two-class classification task, let the desired response t can take two different values such that $t \in \{-1, 1\}$ and $f(x, \alpha)$ represent a set of indicator functions in $\{-1, +1\}$ for different values of α . The loss function is

$$L(t, f(x, \alpha)) = \begin{cases} 0 & \text{if } t = f(x, \alpha) \\ 1 & \text{if } t \neq f(x, \alpha) \end{cases} \quad (5.5)$$

In any two-class classifier (either parametric or non-parameteric), the indicator function $f(x, \alpha)$ can be viewed as a signum of a soft-indicator output $g(x, \alpha)$ such that

$$f(x, \alpha) = \text{sign}(g(x, \alpha)) \quad (5.6)$$

where $\text{sign}(\cdot)$ is $+1$ or -1 if the argument is positive or negative respectively.

For example, in a k -nearest neighbor classifier [2, 28], a test sample x is assigned a label $+1$ (i.e., $f(x, \alpha) = +1$) if the number of positive training samples (that is, training samples with $t = +1$) is more than the number of negative training samples (i.e., training samples with $t = -1$) in a neighborhood $\Omega_k(x)$ of x comprising of k nearest neighbors. The corresponding soft-indicator function $g(x, \alpha)$ can be expressed as

$$g(x, \alpha) = \left(\frac{2N_+(x)}{k} - 1 \right) \quad (5.7)$$

where $N_+(x)$ is the number of positive training samples in $\Omega_k(x)$. We scaled the soft-indicator function $g(x, \alpha)$ such that $g(x, \alpha) \in [-1, +1]$, although it can be scaled differently. The model α here is defined by k . In a decision tree [29, 30], a leaf node is assigned a positive or negative label depending

on the majority of the labels of the training samples allocated to that leaf node. Thus if a test sample x is allocated to a leaf node $l_T(x)$ of a decision tree T then the soft-indicator function $g(x, \alpha)$ is given as

$$g(x, \alpha) = \left(\frac{2N_+(l_T(x))}{N_+(l_T(x)) + N_-(l_T(x))} - 1 \right) \quad (5.8)$$

where $N_+(l_T(x))$ and $N_-(l_T(x))$ are respectively the number of positive and negative training samples allocated to the leaf node $l_T(x)$. In the case of support vector machines (and in various forms of kernel machines), the soft-indicator function $g(x, \alpha)$ is defined as

$$g(x, \alpha) = \sum_i \alpha_i K(x, x_i) t_i + b \quad (5.9)$$

In the case of parametric classifiers such as logistic regression models [31] and semi-parametric classifiers such as neural networks [32] and hierarchical mixture of experts [33], the classifiers themselves produce the soft-indicator function $g(x, \alpha)$. The final outcome of such classifiers is interpreted as $f(x, \alpha) = \text{sign}(g(x, \alpha))$. In short, the outcome of any two-class classifier can be viewed as a soft-indicator function $g(x, \alpha)$ which is then mapped onto $f(x, \alpha) \in \{-1, +1\}$ for making a decision.

The nature of $g(x, \alpha)$ is driven by the underlying model α and the generalization performance of the classifier depends on the nature of $g(x, \alpha)$. In this paper, we refer to $g(x, \alpha)$ as the “outcome” of a classifier. According to the principle, we fix a margin (regularization parameter) λ and minimize the superfluity in the outcome $g(x, \alpha)$ with the given margin. If the outcome $g(x, \alpha)$ is greater than λ for a training sample x with a label $+1$ or if $g(x, \alpha) < -\lambda$ for a training sample x with label -1 then we say that there is superfluity in the outcome since $g(x, \alpha) = \pm\lambda$ is sufficient to make a decision for a given margin λ . For a training sample x with a class label $+1$ ($t(x) = +1$) there is an empirical error in matching the class label if $g(x, \alpha) < \lambda$. Similarly, there is an empirical error if $g(x, \alpha) > -\lambda$ and $t(x) = -1$. We can express the empirical error as

$$L_1(x, t) = \begin{cases} h_1((\lambda t - g(x, \alpha))t) & \text{if } (g(x, \alpha) < \lambda) \text{ and } (t = 1) \\ & \text{or } (g(x, \alpha) > -\lambda) \text{ and } (t = -1) \\ 0 & \text{otherwise} \end{cases} \quad (5.10)$$

where $h_1(\cdot)$ is a monotonically increasing function. Due to superfluity, the error for x is

$$L_2(x, t) = \begin{cases} h_2((g(x, \alpha) - \lambda t)t) & \text{if } (g(x, \alpha) > \lambda) \text{ and } (t = 1) \\ & \quad \text{or } (g(x, \alpha) < -\lambda) \text{ and } (t = -1) \\ 0 & \text{otherwise} \end{cases} \quad (5.11)$$

where $h_2(\cdot)$ is a monotonically increasing function.

We minimize the error due to superfluity and the empirical error together such that the loss functional can be written as

$$L(x, t) = L_1(x, t) + L_2(x, t) \quad (5.12)$$

The parameter λ can be connected to regularization such that if λ is very small then a small perturbation in x can cause $g(x, \alpha)$ to be mapped from one class to the other. Thus for better performance on the training samples, we need to select as large λ as possible. On the other hand, if we select a very large λ then the model α will be highly perturbed by the noisy training samples resulting in a poor generalization performance.

5.2.2. Bias-Variance Perspective

As articulated by [5] (also available in [2, 34, 35]), the overall probability of classification error can be expressed as

$$P[f(x, \alpha) \neq t] = (1 - 2P[t_B \neq t])P[g(x, \alpha) \neq t_B] + P[t_B \neq t] \quad (5.13)$$

where t is the actual observed outcome (in terms of ± 1) for a sample x , t_B is the outcome produced by the optimal Bayes classifier. The second term in Equation (5.13) is the noise component and independent of the classifier. The first term is the product of the error committed by the classifier with respect to the optimal Bayes classifier and a noise term which attenuates the total error of the classifier. If the Bayes classifier output is exactly the same as the actual outcome then the resultant error is the same as that committed by the classifier. However, if the outcome of Bayes classifier deviates from the actual outcome then it imposes a wrongly right effect since $f(x)$ can differ from t_B and t_B can differ from t and therefore $f(x)$ produces the same outcome as t . The optimal Bayes classifier is obtained as

$$t_B = \text{sign}(F(x) > l) \quad (5.14)$$

where

$$F(x) = P[t = 1|x] = 1 - P[t = -1|x] \quad (5.15)$$

The error of the classifier with respect to the optimal Bayes classifier is expressed as

$$P[f(x, \alpha) \neq t_B] = \mathbf{1}(F(x) < l) \int_0^\infty p(g)dg + \mathbf{1}(F(x) > l) \int_{-\infty}^0 p(g)dg \quad (5.16)$$

where $\mathbf{1}(\cdot)$ indicates a 1/0 function where $\mathbf{1}(y) = 1$ if $y > 0$ and 0 otherwise. Intuitively the classifier commits an error when the optimal Bayes classifier assigns a class +1 and the classifier assigns $f(x, \alpha) = -1$, that is, $g(x, \alpha) \leq 0$, and vice-versa. The parameter l is the decision boundary for the Bayes classifier such that

$$l = \frac{c_{-1,1}}{c_{-1,1} + c_{1,-1}} \quad (5.17)$$

where c_{ij} is the cost of misclassifying a sample in class i to class j . As provided in [5], $p(g)$ is assumed to be Gaussian, and the resulting error committed by the classifier with respect to the optimal Bayes model is given as

$$P[f(x, \alpha) \neq t_B] = \Phi[\text{sign}(F(x) - l)\mu_g\sigma_g^{-1}] \quad (5.18)$$

where

$$\mu_g = \mathcal{E}_{\mathcal{D}}[g(x, \alpha; \mathcal{D})] \quad (5.19)$$

is the expectation of $g(x, \alpha)$ over the dataset \mathcal{D} . Similarly, σ_g is the standard deviation or the square-root of the variance of $g(x, \alpha)$ computed over the dataset \mathcal{D} . The function Φ is a nonlinear function given as

$$\Phi(y) = \frac{1}{2}[1 - \text{erf}(y/\sqrt{2})] \quad (5.20)$$

The bias term is dictated by μ_g and the resulting classification bias increases as μ_g decreases. On the other hand, the classification variance decreases as σ_g decreases. The bias and variance are related by certain nonlinear way in the case of classification. As defined by Friedman [5], the bias

$$\text{bias} = \text{sign}(l - F(x))\mu_g \quad (5.21)$$

does not affect the classification so long as the bias is negative. The classification becomes incorrect when the bias becomes positive. This bias is called the boundary bias in classification. The variance is more important aspect of classification and even if the reduction of variance causes high bias, the classification remains correct so long as the boundary bias remains negative. Reduction in the variance and subsequent increase in the bias leads to over smoothing of the classification boundary.

Instead of considering unimodal Gaussian on $g(\cdot)$ if we consider bimodal distribution (mixture of two Gaussians) over $g(\cdot)$ for a two-class problem then the resultant error committed by the classifier with respect to the optimal Bayes model is given as

$$P[f(x, \alpha) \neq t_B] = k_1 \Phi[\text{sign}(F(x) - l)\mu_1\sigma_1^{-1}] + k_2 \Phi[\text{sign}(l - F(x))\mu_2\sigma_2^{-1}] \quad (5.22)$$

where $k_1 + k_2 = 1$ are the weights of each individual Gaussian in the mixture, μ_1 and $-\mu_2$ are the means of individual Gaussians, and σ_1, σ_2 are the respective standard deviations. So long as the sign of $(F(x) - l)$ is consistent with that of μ_1 and sign of $(l - F(x))$ is consistent with $-\mu_2$, the classification is correct. The accuracy increases with the reduction of σ_1 and σ_2 .

In our approach, we also perform the variance reduction by minimizing the superfluity in the outcome. In other words, from Equations (5.10) and (5.11), we observe that the labels t are multiplied by a factor λ which essentially shrinks $g(x, \alpha)$ and thus the variance is reduced. However, if we simply use linear scaling of the model $g(x, \alpha)$ then the variance reduction is completely cancelled by the reduction of mean (gain in bias) and there is effectively no change in the classification performance. If we identify the model parameters within the same setting (without any linear scaling) to restrict the superfluity in the outcome then the reduction of variance is not cancelled by the gain in bias. In the sequel, we show the use of this principle to design two different classifiers where we gain in terms of variance with respect to certain parameter λ .

5.2.3. Relationship with the Regularization using Model Complexity

As a special case, if we restrict $h_1(u) = h_2(u) = u^2$ then the resultant loss functional for a training sample x (Equation (5.12)) becomes

$$L(x, t) = (g(x, \alpha) - \lambda t(x))^2 \quad (5.23)$$

Thus the total normalized loss over the training dataset \mathcal{D} is

$$L(\mathcal{D}|\alpha) = \frac{1}{|\mathcal{D}|} \sum_x (g(x, \alpha) - \lambda t(x))^2 \quad (5.24)$$

The Equation (5.24) can be expressed as

$$L(\mathcal{D}|\alpha) = \frac{1}{|\mathcal{D}|} \sum_x [\lambda(g(x, \alpha) - t(x))^2 + (1 - \lambda)g^2(x, \alpha) - \lambda(1 - \lambda)t^2(x)] \quad (5.25)$$

Since the third term in Equation (5.25) is independent of α , we can equivalently express $L(\mathcal{D}|\alpha)$ as

$$L(\mathcal{D}|\alpha) = \frac{1}{|\mathcal{D}|} \sum_x (g(x, \alpha) - t(x))^2 + \frac{1-\lambda}{\lambda} \frac{1}{|\mathcal{D}|} \sum_x g^2(x, \alpha) \quad (5.26)$$

The second term in Equation (5.26) imposes explicit regularization on the outcome of the classifier and $\frac{1-\lambda}{\lambda}$ is the Lagrangian multiplier of the regularization functional. Since $\frac{1}{|\mathcal{D}|} \sum_x g^2(x, \alpha)$ is a function of α , we can equivalently express Equation (5.26) as

$$L(\mathcal{D}|\alpha) = \frac{1}{|\mathcal{D}|} \sum_x (g(x, \alpha) - t(x))^2 + \gamma \Gamma(\alpha) \quad (5.27)$$

where $\gamma = (1 - \lambda)/\lambda$ is the Lagrangian multiplier and

$$\Gamma(\alpha) = \frac{1}{|\mathcal{D}|} \sum_x g^2(x, \alpha) \quad (5.28)$$

is a function of the model parameters α ; $\Gamma(\alpha)$ represents the model complexity. Thus although we express the principle in terms of reducing the superfluity in the outcome, it is connected to the explicit regularization of the model complexity. However, the model complexity in Equation (5.27) is not the same as that defined in MDL or SRM. For example, in SRM principle, the model complexity is driven by the worst-case performance over the training samples whereas in the present framework the model complexity is the average over all training samples. Often the squared norm of the model parameter vector is considered in the Tikhonov regularization. In our approach, we do not explicitly consider any function $\Gamma(\cdot)$ for regularization, rather the regularization function is derived from the outcome of the classifier itself. Also we observe that the contribution of the regularization term increases as we decrease λ and effectively bias increases. On the other hand for $\lambda = 1$, the regularization term vanishes.

5.3. Design of Classifiers using the Principle of Parsimony

In this section, we design two different classifiers using the stated principle.

5.3.1. A Kernel Machine

We consider a kernel based classifier such that the function $g(x, \alpha)$ is given as

$$g(x, \alpha) = \sum_j \alpha_j K(x, x_j) t_j \quad (5.29)$$

where x_j is an observed sample and t_j is the given label of the observed sample, and α_j indicates the contribution of the sample j in the classifier. $K(x, x_j)$ is a kernel function such as Gaussian kernel of the form

$$K(x, x_j) = \exp\left(-\frac{\|x - x_j\|^2}{2\sigma^2}\right) \quad (5.30)$$

The kernel is not necessarily restricted to Gaussian kernels. We restrict to $h_1(u) = h_2(u) = u^2$ (Equation (5.12)) and therefore the resultant loss functional (not normalized by the number of training samples) is given as

$$L = \frac{1}{2} \sum_i \left(\sum_j \alpha_j K(x_i, x_j) t_j - \lambda t_i \right)^2 \quad (5.31)$$

Using unrestricted parameter values, we can minimize L with respect to α using linear regression technique and obtain one classifier for one λ . In deriving the parameter values $L(\alpha)$ can be differentiated with respect to α and we obtain exactly $|\mathcal{D}|$ equations such that we can obtain an exact solution for α . However, the solution for α is simply linear in λ , and therefore λ linearly scales $g(x, \alpha)$. As discussed in Section 5.2.2, if we simply perform linear scaling of $g(x, \alpha)$ then the gain in the variance reduction is completely cancelled by the increase in bias. Therefore if we use unconstrained linear regression then for any value of λ we obtain the same classifier as obtained by minimizing the empirical error.

In order to effectively gain in variance reduction, we constrain the parameters in $0 \leq \alpha \leq 1$. Thus for a given λ , the minimization of L with respect to α is equivalent to maximization of the functional

$$W(\alpha) = \lambda t^T K D(t) \alpha - \frac{1}{2} \alpha^T D(t) K^T K D(t) \alpha \quad (5.32)$$

subject to the condition that $0 \leq \alpha_i \leq 1$ for all i . The entry $D(t)$ is a diagonal matrix whose diagonal entries are the same as that of t . Once we maximize the functional $W(\cdot)$ with respect to α we obtain certain non-zero α s which define the classifier. Since we constrain $0 \leq \alpha \leq 1$, as we increase λ , maximization of the objective functional $W(\cdot)$ generates new non-zero vectors which changes the nature of the distribution. The introduction of new non-zero vectors causes an additional increase in the variance. Thus by extending this logic, we notice that decrease in λ reduces μ_g causing an increase in the bias, whereas the variance is decreased further (decrease in the number of non-zero vectors). If we decrease λ towards zero then it introduces infinite bias and no variance, and there is no classification (all the samples are treated as same). Therefore, for a particular classification task, there exist an optimal range of λ over which the classifier provides an

optimal performance. If two classes are completely separable then we can obtain a wide range of λ whereas for a highly overlapped class structure, we can identify a specific narrow range of λ over which the classifier performs optimally.

Once the pattern vectors with non-zero α values are obtained by maximizing the functional $W(\alpha)$ with respect to α_i s (Equation (5.32)), we classify any new sample x , i.e., assign a class label t to x as

$$t = \begin{cases} 1 & \text{if } \sum_i \alpha_i K(x, x_i) t_i \geq 0 \\ -1 & \text{otherwise} \end{cases} \quad (5.33)$$

Multi-class Classification: We formulated the classification problem for a two-class classification task. In the case of multi-class classification, we consider the one-against-all strategy, that is, classify the patterns of each class against all other classes and obtain the model coefficients α for each class separately. Here we mention that we take the stand-point as provided in [36] where the authors have shown that one-against-all classification scheme is as accurate as any other multi-class classification approach.

Formally let there be l class labels such that the set of class labels is denoted by $CL = \{1, 2, \dots, l\}$. Each time we consider one class label at a time. Let the class label of concern be $c \in CL$. In that case $t_i \in CL$ is transformed into a vector $\{t_{i1}, t_{i2}, \dots, t_{il}\}$ such that

$$t_{ic} = \begin{cases} 1 & \text{if } t_i = c \\ -1 & \text{otherwise} \end{cases} \quad (5.34)$$

For each class label c , we compute the coefficients separately, and assign the class label t to x as

$$t = \operatorname{argmax}_{c \in L} \left\{ \sum_i \alpha_{ic} K(x, x_i) t_{ic} \right\} \quad (5.35)$$

where α_{ics} are the coefficients obtained for class c .

5.3.1.1. Interpretation of the Kernel Machine

As we mentioned in Section 2.3, the objective functional for the kernel machine can be expressed as a combination of the empirical error and the model complexity in the form

$$L(\mathcal{D}|\alpha) = \frac{1}{|\mathcal{D}|} \sum_x (g(x, \alpha) - t(x))^2 + \gamma \Gamma(\alpha) \quad (5.36)$$

where γ is the regularization parameter ($\gamma = (1 - \lambda)/\lambda$) and $\Gamma(\alpha)$ is the model complexity given as

$$\Gamma(\alpha) = \frac{1}{|\mathcal{D}|} \sum_x g^2(x, \alpha) \quad (5.37)$$

Considering

$$g(x, \alpha) = \sum_j \alpha_j K(x, x_j) t_j \quad (5.38)$$

the model complexity is given as

$$\Gamma(\alpha) = \alpha^T D(t) K^T K D(t) \alpha \quad (5.39)$$

In other words, we minimize a quadratic loss functional equivalent to the empirical error in addition to a regularization function governed by the model complexity; the model complexity being dependent on the outcome of the classifier.

As we impose the constraints on the priors α , the loss can be reformulated as

$$L = \frac{\lambda^2}{2} \sum_i \left(\sum_j \frac{\alpha_j}{\lambda} K(x_i, x_j) t_j - t_i \right)^2 \quad (5.40)$$

which is equivalent to minimizing a functional

$$L = \frac{1}{2} \sum_i \left(\sum_j K(x_i, x_j) \hat{\alpha}_j t_j - t_i \right)^2 \quad (5.41)$$

subject to the constraints $0 \leq \hat{\alpha}_i \leq \frac{1}{\lambda}$. From the Equation (5.41) and the respective constraints, we observe the kernel machine employs a uniform prior over the coefficients α . The uniform prior can be better observed if we replace $\hat{\alpha}_i t_i$ by β_i then we obtain the quadratic loss

$$L = \frac{1}{2} \sum_i \left(\sum_j K(x_i, x_j) \beta_j - t_i \right)^2 \quad (5.42)$$

subject to the constraint that $0 \leq |\beta_i| \leq \frac{1}{\lambda}$, and $\text{sign}(\beta_i) = t_i$ where t_i is a binary observation in $\{-1, 1\}$. In other words, we minimize a quadratic loss functional equivalent to the empirical error subject to the box constraints on the model coefficients; the size of the box depends on the parameter λ (the margin according to the principle).

Alternatively combining Equations (5.36) and (5.42), we can express

$$L = \|t - K\beta\|^2 + \gamma \|K\beta\|^2 \quad (5.43)$$

With normalization,

$$L = \frac{1}{|\mathcal{D}|} (t - K\beta)^T (t - K\beta) + \frac{\hat{\gamma}}{2} \beta^T K^T K \beta \quad (5.44)$$

where $\beta \in [-1, 1]$ and $\beta_i t_i \geq 0$ and $\hat{\gamma} = \frac{2(1-\lambda)}{|\mathcal{D}| \lambda}$. Thus the coefficients have uniform prior over $[0, \pm 1]$ depending on the class label of the observed sample and regularized by the model complexity with a regularization parameter γ or $\hat{\gamma}$. We express the objective functional in this form in order to show the similarity with other existing kernel machines in the next section.

5.3.1.2. Relationship with Other Kernel Machines

The effects of α s are similar to that used in the SVM where α s define the support vectors. However the quadratic function to be maximized in SVM is derived on the basis of margin maximization with respect to optimal separating hyperplane in some higher dimensional space $\phi(\cdot)$ such that K is expressible as a inner product, $K(x, x_i) = \phi(x)\phi(x_i)$, and α s are the Lagrangians. In our case, we do not explicitly consider the margin maximization with respect to separating hyperplane in the space of $\phi(\cdot)$, rather we minimize the outcome for a given parameter λ .

This leads to a difference in the quadratic term $K^T K$ in our formulation in Equation (5.32) from that in the SVM. Apart from the quadratic term, we also observe the difference with SVM in the linear term. In SVM, due to margin maximization all α s contribute equally in the linear term as in

$$W_{svm}(\alpha) = \mathbf{1}^T \alpha - \frac{1}{2C} \alpha^T D(t) K D(t) \alpha \quad (5.45)$$

with an additional constraint that $\alpha^T t = 0$ and $\alpha_i \geq 0$ for all i . In the modified framework of soft margin SVM [37, 38], the functional is expressed as

$$W_{softsvm}(\alpha) = \mathbf{1}^T \alpha - \frac{1}{2} \alpha^T D(t) (K + \frac{1}{C} I) D(t) \alpha \quad (5.46)$$

where I is the identity matrix with the same constraint i.e., $\alpha^T t = 0$. We do not require this constraint (Equation (5.32)) since different α s contribute differently in the linear term depending on the kernel function and the distribution of the patterns.

In the rest of this paper, we refer to our classifier as the *Least Square Kernel Machine with Box Constraints (LSKMBC)* since we are modeling the optimization framework within a box [39, 40]. The non-zero α s determine the *Box vectors* or simply *B-vectors* as opposed to the support

vectors. Since in the formulation of the kernel based classifier, we do not use the principle of margin maximization, we do not claim this as a variant of SVM.

As we view the expanded quadratic loss in Equation (5.32), we observe the similarity of LSKMBC with ν -SVM [41], where the optimization functional is given as

$$L = -\frac{1}{2}\alpha^T D(t) K D(t)\alpha \quad (5.47)$$

subject to $0 \leq \alpha_i \leq \frac{1}{\lambda}$, λ being a regularization parameter $\lambda > 0$, and $\alpha^T t = 0$, $\sum \alpha_i \geq \nu$. In ν -SVM the quadratic optimization in ν -SVM does not contain any linear term as in standard SVM. The performance of ν -SVM [41] depends on two parameters namely ν and λ . ν -SVM uses a principle of regularizing the margin of separation between the classes (with an additional regularization factor) where the parameter ν controls the penalization due to separation of the classes in the primal formulation of the loss functional. The separation between the classes is governed by the separation margin ρ and a regularization functional $-\nu\rho$ is added to the primal formulation. The minimization of the loss functional is subjected to the maximization of the separation margin ρ and the effect of ρ depends on the parameter ν . In this respect, LSKMBC also uses a parameter λ to control the separation between the classes in terms of the outcome of the classifier. However, the primal formulation of ν -SVM is derived from the margin maximization principle as in the SVM whereas in LSKMBC, we do not explicitly consider the margin maximization as a primal formulation; however, we consider the minimization of the loss functional with a given margin (regularization parameter) directly in its dual form.

In the ridge regression framework [42], the least square regression includes a regularization factor such that

$$L = \|t - w^T x\|^2 + a\|w\|^2 \quad (5.48)$$

where a is a Lagrangian. The ridge regression framework introduces priors over the coefficients w for a smooth regression function estimation, the priors being subjected to a spherical Gaussian prior distribution. In adaptive ridge regression [43], automatic relevance determination is performed in such a way that each variable is penalized by the method of automatic balancing while keeping the average penalty constant. In RLSC [44], the objective functional is derived as a simple square-loss function and regularized using the Tikhonov regularization [22] with a quadratic term in α .

vectors. In RLSC reproducing Hilbert kernels are considered such that the objective functional takes a form

$$L = \frac{1}{N}(t - K\alpha)^T(t - K\alpha) + \frac{\gamma}{2}\alpha^T K\alpha \quad (5.49)$$

where N is the number of samples, γ is a regularization parameter, and α are the coefficients (the notation of α is same throughout this article). This convex optimization problem is then transformed into a set of linear equations (similar to least square regression) by equating the first order derivative to zero based on the assumption that the underlying noise is generated by a Gaussian model. The coefficients are then derived from the resulting set of linear equations such that

$$(K + \gamma NI)\alpha = t \quad (5.50)$$

In least-square support vector machine (LSSVM) [45, 46], a squared error term is considered in addition to a regularization term w^Tw where w represents the directionality of the separating hyperplane. The convex optimization functional is then transformed into a set of linear equations by equating the first derivative to zero using unconstrained priors α .

We observe that the LSKMBC as expressed in Equations (5.43) and (5.44) is similar to the adaptive ridge regression (AdR) and RLSC. However, in AdR, RLSC, and LSSVM, the form of model complexity that has been used to regularize the model parameters is different from that in the LSKMBC. Therefore minimization of L with unconstrained priors (α) in such models does not yield simple scaling of the outcome $g(x, \alpha)$. In the LSKMBC on the other hand, unconstrained minimization results into simple scaling and we constrain the the parameters α . Moreover, RLSC and other related family of kernel machines are restricted to Hilbert kernels subjected to Mercer condition which is relaxed in the LSKMBC. The LSKMBC is not limited only to Hilbert kernels since the quadratic term in Equation (5.32) is always positive semi-definite.

Constrained priors have been used in LASSO (least absolute shrinkage and selection operator) [47, 48], and generalized LASSO [31, 49]. In LASSO, the loss functional considered is given as

$$L = \|t - K\alpha\|_2^2 \quad (5.51)$$

subject to $\|\alpha\|_1 < \lambda$ where λ is a constant which determines the approximation accuracy. The LASSO model is very similar to the LSKMBC as interpreted in Equation (5.42) except that the constraints over the priors

are different. Considering the exponential hyperpriors, the LASSO model can be expressed in the form

$$L = \|t - K\alpha\|_2^2 + b\|\alpha\|_1 \quad (5.52)$$

where b is a Lagrange parameter determined by the exponential hyperpriors. The generalized LASSO uses a very similar form of loss functional as in Equation (5.52) except that it uses a more robust form of Huber loss [50] which is quadratic for smaller deviation and linear for larger deviations. Subsequently iteratively reweighted least square (IRLS) technique is used to optimize the functional. The constraints that we use in the LSKMBC (Equation (5.43)) are different from the constraints used in LASSO and generalized LASSO.

In relevance vector machines [51], empirical error is defined based on the logistic regression model where the outcome is given as $g(x) = 1/(1 + \exp(-w^T \phi(x)))$. In a Bayesian framework zero-mean Gaussian priors are defined over the parameter vectors w such that $p(w|\alpha) = \prod_i \mathcal{N}(0, 1/\alpha_i)$ where \mathcal{N} represents the Gaussian distribution. In this framework, an iterative method is employed. It starts with a current estimate of α vectors, and based on this estimate the most probable hyperprior (prior variance) parameters are chosen. The prior variance parameters are then used in the posterior to get new estimates of α . Thus relevance vector machine explicitly regularizes the model parameters w within a Bayesian framework with respect to the priors α and iteratively estimates the priors. In the LSKMBC, we do not use the logistic regression model for classification for measuring the empirical error and instead of Gaussian hyperpriors, we use uniform priors over the parameters α .

5.3.2. A Modified k -Nearest Neighbor Algorithm

As we discussed in Section 2.1 (Equation (5.7)) that the outcome for a test samples can also be expressed as a soft indicator function, we apply the proposed principle to design a modified k -nearest neighbor classifier using the soft indicator function. In the case of k -NN classifier, the model parameter is k or the number of nearest neighbors being considered. Therefore according to this principle, we decide the value of k for every test sample with a given value of λ . On the contrary, in the standard k -NN classifier, all test samples are classified for a given k (the value of k can be decided by cross-validation). First, we scale the outcome of the classifier for a test sample in $[0, 1]$ for l different class labels. We assign a function $g(x, k)$ such

that

$$g(x, k) = \frac{lN_t(x) - k}{k(l-1)} \quad (5.53)$$

where $N_t(x) > N_i(x)$ for all $i \neq t$ and $i, t \in CL$. $N_i(x)$ is the number of training samples with a class label i in the k nearest neighboring training samples of x and $CL = \{1, 2, \dots, l\}$ is the set of class labels. We scale the outcome for a test samples in $[0, 1]$ so that the value of λ can be chosen in the range $[0, 1]$ irrespective of the number of class labels. We then compute the outcome for the test sample x for a large range of k and choose that particular k for which $g(x, k)$ is closest to λ and assign the respective class label to x . Formally, the algorithm is presented in Figure 5.1. In this

```

Input: Training dataset ( $\mathcal{D}$ ); Test dataset;
Output: class label  $t(x)$  for every test sample  $x$ ;
Input Variable:  $\lambda$ ;
begin
  for every test samples  $x$  do
    begin
      for  $k := 1$  to  $k_{max}$  do
        begin
          Compute the class label  $f(x, k) = t$  if  $N_t(x) > N_i(x)$ 
          for all  $i \neq t$ ,  $i, t \in CL$ 
          Compute the outcome  $g(x, k) = \frac{lN_t(x)-k}{k(l-1)}$ 
        end
        Find  $k_{opt} = \operatorname{argmin}_k |g(x, k) - \lambda|$ 
        Assign class label to  $x$  such that  $t(x) = f(x, k_{opt})$ 
      end
    end
end

```

Fig. 5.1. Modified k -nearest neighbor classifier (λ -kNN).

algorithm, we do not use any kernel function to implement the k -nearest neighbor algorithm. Rather the model parameter k is decided from the outcome of the classifier for a given margin λ . Since the performance of this classifier is solely dependent on the parameter λ , we refer to this classifier as λ -kNN classifier in the sequel. Interestingly, we note that the value of k in λ -kNN can differ for different test samples. If the classifier performs best of $k = 1$ for all test samples (that is, 1-nearest neighbor classifier) then the value of optimal λ will be unity.

5.4. Experimental Results

5.4.1. Nature of the LSKMBC

We first illustrate the B-vectors generated by LSKMBC for two-dimensional Gaussian parity problem. We select the Gaussian kernel of different sizes. First, in Figure 5.2, we illustrate the B-vectors for $2\sigma^2 = 1$. The B-vectors are marked by ‘o’, and the samples from two different classes are marked by ‘x’ and ‘.’ respectively. We observe that even though we do not perform margin maximization as performed in SVM, the B-vectors concentrate towards the boundary between opposite classes. However, this behavior of the B-vectors is specific to the choice of kernel. In the next figure, we observe a different behavior of the B-vectors as we change the kernel width.

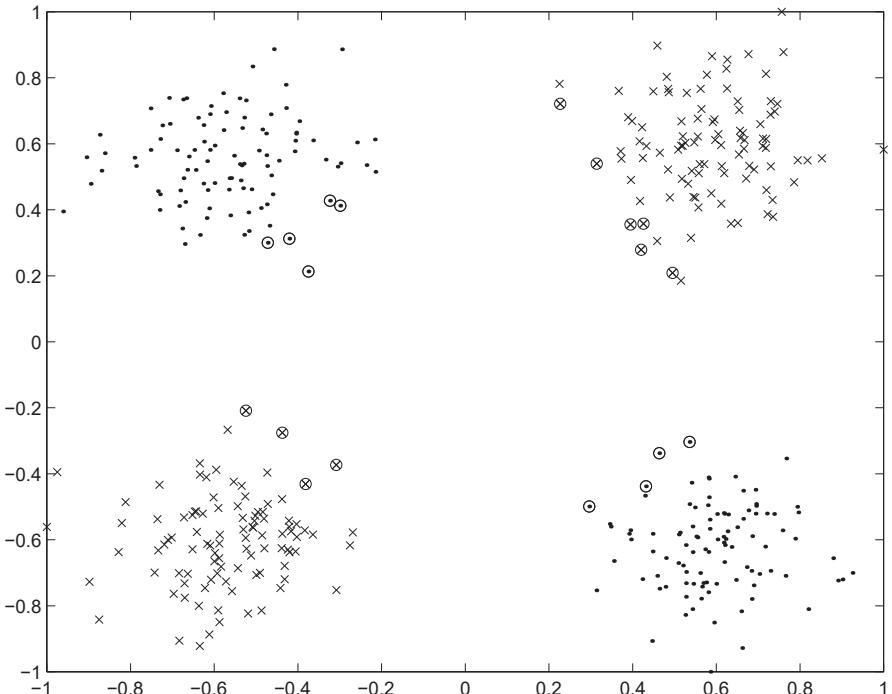


Fig. 5.2. B-vectors obtained by the LSKMBC for a two-dimensional Gaussian parity problem. The two classes are represented by ‘x’ and ‘.’ respectively. The B-vectors are marked by ‘o’. The Gaussian kernel of the LSKMBC is chosen such that $2\sigma^2 = 1$.

Figure 5.3 illustrates the change in the B-vectors as we change σ of Gaussian kernel as 0.2, 0.5, 1 and 2. We observe that for a low σ , the B-vectors are not necessarily concentrated near the opposite classes. This is due to the fact that LSKMBC does not use the principle of margin maximization, rather it finds B-vectors such that existence of other points becomes interpretable subject to a given margin. As we increase σ , the B-vectors starts getting concentrated towards the opposite class samples. This is due to higher interaction between samples from other classes due to larger width of the kernel function.

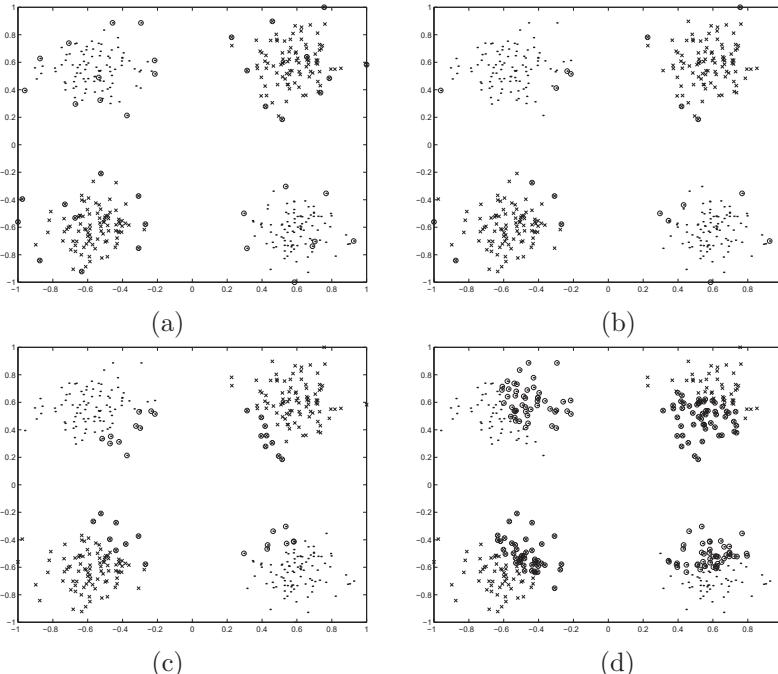


Fig. 5.3. B-vectors obtained by LSKMBC for the same two-dimensional Gaussian parity problem as in Figure 5.2 with different kernel sizes. (a), (b), (c), and (d) illustrate the B-vectors for $\sigma = 0.2$, $\sigma = 0.5$, $\sigma = 1.0$ and $\sigma = 2.0$ respectively.

We now illustrate the effect of λ on the number of B-vectors. If we lower the margin λ then we observe that the number of B-vectors decreases and it increases with the increase in λ . This is due to the fact that if we reduce the margin, we require less number of B-vectors to support the existence of other samples from the same class subject to the margin. Figure 5.4 illustrates the B-vectors for the same Gaussian parity problem with $\lambda = 0.1$, 0.2 , 0.5 , and 1 respectively for $\sigma = 1$.

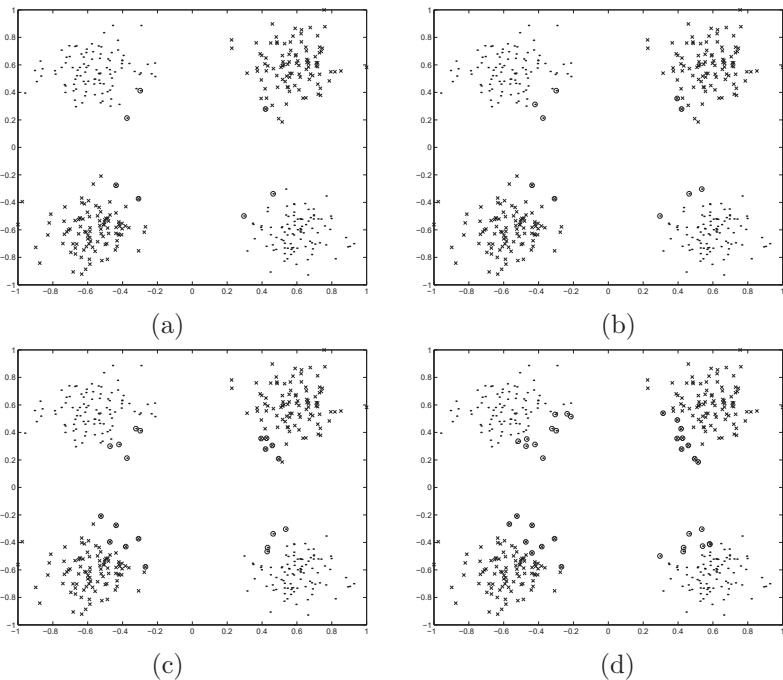


Fig. 5.4. B-vectors obtained by LSKMBC for the same two-dimensional Gaussian parity problem as in Figure 5.2 for different margins (λ) with a fixed size of Gaussian kernel $\sigma = 1$. (a), (b), (c), and (d) illustrate the B-vectors for $\lambda = 0.1$, $\lambda = 0.2$, $\lambda = 0.5$, and $\lambda = 1.0$ respectively.

5.4.2. Data Sets

We demonstrate the effectiveness of our classifier on certain real-life data sets. We have considered the data sets available in the UCI machine learning repository [52]. Table 5.1 summarizes the data set description. We considered the data sets from UCI machine learning repository [52] mostly based on the fact that the data should not contain any missing variable and consist of only numeric attributes. Note that, the ‘Iris’ dataset that we used is the ‘BezdekIris’ data in the UCI repository. We modified the ‘Ecoli’ data originally used in [53] and later in [54] for predicting protein localization sites. The original dataset consists of eight different classes out of which three classes namely, outer membrane lipoprotein, inner membrane lipoprotein, and inner membrane cleavable signal sequence have only five, two, and two instances respectively. Since we report the 10-fold cross-

validation score, we omitted samples from these three classes and report the results for the rest five different classes. Note that, in the original work [53] also, the results are not cross-validated. We normalize all input patterns such that any component of \mathbf{x} lies in the range $[-1, +1]$.

Table 5.1. Description of the pattern classification datasets obtained from the UCI machine learning repository.

Data Set	Number of Instances	Number of Features	Number of Classes
Indian diabetes (Pima)	768	8	2
Diagnostic Breast Cancer (Wdbc)	569	30	2
Prognostic Breast Cancer (Wpbc)	198	32	2
Liver Disease (Bupa)	345	6	2
Flower (Iris)	150	4	3
Bacteria (Ecoli)	326	7	5

5.4.3. Performance of the LSKMBC

5.4.3.1. Choice of the Kernel Functions

The kernel functions of the LSKMBC is not necessarily subjected to Mercer condition (Hilbert kernels) since the quadratic term in the objective functional of LSKMBC (Equation (5.32)) is always positive semi-definite. Apart from Gaussian kernels, we select two other kernel functions which are centrally peaked with longer tails in nature, such that the distant B-vectors have greater interaction between. The first one is given as

$$K_1(x, \mu) = \frac{1}{1 + \left(\frac{\|x - \mu\|}{\sigma}\right)^2} \quad (5.54)$$

The kernel in Equation (5.54) is derived from the Cauchy distribution which has a long tail, and we call this kernel function as Cauchy kernel. Note that, Cauchy kernel has also been applied in the formation of sparse codes for natural scenes leading to a complete family of localized, oriented, bandpass receptive fields [55]. Cauchy kernel is continuous and it is differentiable except at the point $x = \mu$.

The second form of kernel is a mixture of Guassian and exponential kernel such that for smaller deviation it is squared loss and for larger deviation the loss is linear. In the second kernel function, we use Gaussian kernel near the center of the kernel (the logarithm of Gaussian is essentially squared deviation) and exponential decay away from the center (the

logarithm of exponential represents linear deviation). The second kernel function is given as

$$K_2(x, \mu) = \begin{cases} \exp\left(-\frac{\|x-\mu\|^2}{2\sigma^2}\right) & \text{for } \|x-\mu\| \leq \epsilon \\ \exp\left(-\frac{\|x-\mu\|}{\sqrt{2}\sigma}\right) & \text{otherwise} \end{cases} \quad (5.55)$$

Note that, K_2 is continuous only for $\epsilon = \sqrt{2}\sigma$ and we choose the same value although in the framework of LSKMBC it is not necessary that the kernel function needs to be continuous. We refer to the kernel K_2 as the *Gaussian + Exponential* kernel which is not differentiable on the hyperplane $\|x-\mu\| = \sqrt{2}\sigma$. In the next section, we demonstrate the performance of LSKMBC with *Cauchy* kernel and *Gaussian + Exponential* kernel in addition to *Gaussian* kernel.

5.4.3.2. Results

We report the 10-fold cross-validation performance of LSKMBC for Gaussian kernel, Cauchy kernel, and Gaussian+Exponential kernel respectively. In computing the 10-fold cross-validation scores, we randomly partitioned the data into 10 disjoint groups, and used 90% (i.e., 9 disjoint groups) samples as the training sample and the rest 10% samples as the test samples, and iterated this procedure using each group in the partition as the test set and the rest as the training set. We used 10 such trials of random partitioning the data. For each trial, we compute the mean and variance of the classification score over the ten different test sets, and then compute the overall mean and variance scores over the ten different trials. We performed this random partitioning of the dataset and averaged over ten different trials in order to reduce any inherent bias generated due to sample ordering present in the data set. We also compare the performance of LSKMBC with that of SVM using Gaussian kernel and third degree polynomial kernels. In addition we compare the performance with the LSSVM using Gaussian kernels. In comparing the performance, we use the same training set and the test set for each trial and each fold throughout for all these classifiers.

Table 5.2 summarizes the 10-fold cross-validation score of LSKMBC on the datasets in Table 5.1 for Gaussian kernels, Cauchy kernels, and Gaussian+Exponential kernels. We report the mean accuracy as well as the standard deviation in the accuracy as the indicator of significance. As a comparison, we also provide the same scores of SVM for Gaussian kernel and polynomial kernel, and LSSVM with Gaussian kernel in Table 5.2. For implementing SVM, we used the publicly available code in [56]. We

implemented LSSVM using the Matlab code available in [57]. In implementing LSSVM for multiclass classification (such as ‘Iris’ and ‘Ecoli’), we used ‘minimum output coding’ [57] which provided the best performance for the LSSVM. In Table 5.2, we report the performances of all the classifiers namely, SVM, LSSVM, and LSKMBC. In order to test the performance, we divide the dataset into 10 folds. We then consider each 10We then report the performance as the average of these 10 folds. Within each fold we select the parameter values by further cross-validation of the training data to remove any bias.

Table 5.2. Classification performance in terms of the average 10-fold scores on the datasets described in Table 5.1. Scores summarize the mean accuracies and corresponding standard deviations.

Classifier	Pima	Bupa	Wdbc	Wpbc	Iris	Ecoli
SVM (Gaussian)	76.88 (± 4.74)	69.39 (± 6.53)	98.03 (± 1.80)	80.78 (± 6.15)	96.13 (± 4.87)	88.17 (± 4.53)
SVM (Polynomial)	75.58 (± 4.74)	71.72 (± 6.40)	96.20 (± 2.26)	74.39 (± 8.72)	96.13 (± 5.06)	87.25 (± 4.88)
(C)	(1)	(5)	(1)	(1)	(5)	(3)
LSSVM (Gaussian)	76.28 (± 5.18)	68.87 (± 7.21)	93.78 (± 2.93)	78.13 (± 3.93)	84.53 (± 7.90)	77.40 (± 5.54)
LSKMBC (Gaussian)	77.51 (± 5.41)	72.42 (± 6.37)	97.68 (± 1.65)	81.05 (± 7.11)	95.87 (± 5.02)	87.93 (± 4.79)
LSKMBC (Gauss+Expo)	76.28 (± 4.78)	73.17 (± 6.59)	97.61 (± 1.76)	80.83 (± 6.35)	97.07 (± 4.70)	88.77 (± 4.56)
LSKMBC (Cauchy)	77.10 (± 5.40)	72.77 (± 6.55)	97.81 (± 1.65)	82.33 (± 6.51)	97.60 (± 4.21)	89.16 (± 4.37)

As we observe from Table 5.2, LSKMBC outperforms SVM and LSSVM in terms of 10-fold cross-validation score on several datasets. For the ‘Pima’, ‘Bupa’, ‘Wpbc’, ‘Iris’ and ‘Ecoli’ datasets, LSKMBC outperforms SVM and LSSVM, particularly when we observe the performance of LSKMBC with Cauchy kernel. For ‘Wdbc’ dataset, the best performance of SVM is marginally better than that of LSKMBC although LSSVM is much worse than the other two. Interestingly, we observe that the performance of LSKMBC often improves significantly with long-tailed kernels such as Cauchy kernel.

In order to establish the comparison between these classifiers in a more quantitative way, we performed resampled paired t-test as provided in [58]. As described in [58], we randomly divided the dataset such that two-third of the dataset constituted a training set, and the rest one-third constituted a test set. We then used the same training set and test set for all the variants

of the three classifiers. We then computed the rate of misclassification on the test set. We repeated this experiment for 30 different trials, where in every trial we randomly partitioned the data and computed the rate of misclassification by each classifier. For every trial, we computed the difference in the rate of misclassification. For example, if c_1 and c_2 are two different classifiers with rates of misclassification $p_1^{(i)}$ and $p_2^{(i)}$ respectively for trial i , then the difference in misclassification is $p^{(i)} = p_1^{(i)} - p_2^{(i)}$, and the statistic is obtained as

$$t = \frac{\bar{p} \cdot \sqrt{N}}{\sqrt{\frac{\sum_{i=1}^N (p^{(i)} - \bar{p})^2}{N-1}}} \quad (5.56)$$

where $N (= 30)$ is the number of trials, and $\bar{p} = \frac{1}{N} \sum_{i=1}^N p^{(i)}$ is the average difference in the misclassification rate. Evidently, if $t < 0$ then classifier c_1 is better than the classifier c_2 and vice-versa. The significance to which the two classifiers are different is obtained from the measure t . As provided in [58] (as obtained from the Student's cumulative t-distribution function with $N - 1 (= 29)$ degrees of freedom), if $|t| > 2.0452$ then the classifiers are different from each other with a confidence level of 97.5%, and the classifiers are different with a 95% confidence level if $|t| > 1.6991$. Table 5.3 summarizes the t-statistic as obtained by the resampled paired t-test.

Table 5.3. Resampled paired t-test scores in terms of t-statistic comparing the three variants of LSKMBC with the two variants of SVM and LSSVM on the datasets described in Table 5.1.

Classifier Pair	Pima	Bupa	Wdbc	Wpbc	Iris	Ecoli
LSKMBC (Gauss)	-3.94	-7.33	0.21	-3.33	-3.26	2.37
SVM (Gauss)						
LSKMBC (Gauss)	-7.0	-3.81	-6.93	-9.59	-2.72	-2.19
SVM (Poly)						
LSKMBC (Gauss)	-4.27	-8.10	-14.73	-4.75	-17.83	-12.10
LSSVM						
LSKMBC (GaussExpo)	1.73	-8.00	1.48	-0.41	-5.43	-2.54
SVM (Gauss)						
LSKMBC (GaussExpo)	-2.03	-4.61	-4.91	-6.41	-4.63	-8.53
SVM (Poly)						
LSKMBC (GaussExpo)	1.52	-8.56	-14.11	-2.29	-17.81	-13.34
LSSVM						
LSKMBC (Cauchy)	-4.84	-6.97	1.09	-2.71	-4.82	-3.25
SVM (Gauss)						
LSKMBC (Cauchy)	-7.4	-3.49	-5.38	-7.45	-4.85	-6.56
SVM (Poly)						
LSKMBC (Cauchy)	-4.9	-7.91	-14.68	-3.90	-17.94	-13.79
LSSVM						

In ‘Wdbc’ dataset, the best variant of LSKMBC (Gaussian kernel) is worse than the best SVM (Gaussian kernel) with a confidence 58.24% ($t = 0.21$), however, the LSKMBC performs significantly better (with a probability very close to unity) than LSSVM. In all other datasets, we observe that the best variant of LSKMBC significantly outperforms the best variant of SVM and LSSVM.

5.4.3.3. Adult and Web Datasets

In Platt [59], two different datasets namely ‘adult’ and ‘web’ datasets were used. In the adult dataset, the task is to classify households whether having annual income greater than USD 50K or not based on the 14 different census fields including eight categorical variables. The data is transformed into 123 sparse binary attributes where six continuous variables are quantized into quantiles. In the web dataset, the task is to predict whether a web page belongs to particular category or not based on the presence of 300 different keywords. Thus the web dataset also has 300 sparse binary attributes. The original ‘adult’ and ‘web’ datasets consist of 32562 and 49749 training samples. Since we used Matlab quadratic programming library to directly optimize the objective functional of the LSKMBC, the large number of samples could not be accommodated due to the limitation of the virtual memory. Originally Platt [59] used sequential minimal optimization for the larger datasets, however, we have not used any equivalent implementation of the LSKMBC. Platt [59] used nested sets of training samples for both the ‘adult’ and the ‘web’ datasets. We used the first two subsets of the ‘adult’ dataset and the first subset of the ‘web’ dataset. In Table 5.4, we show the respective details of the ‘adult’ and ‘web’ datasets that we used for the experimentation. These datasets are highly unbalanced (samples from one class largely dominate the dataset) and SVM is able to perform classification with high accuracy even for these unbalanced datasets. We compare the performance of the LSKMBC with that of SVM and LSSVM for different types of kernels. In Table 5.5 we report the results over the test samples for all classifiers with the best parameter settings. Here also we observe that the LSKMBC performs better than the SVM and LSSVM although the differences in performance scores are not so significant. However this shows that the LSKMBC is able to perform well even for the unbalanced datasets such as adult and web datasets.

Table 5.4. Description of the nested subsets of the ‘adult’ and ‘web’ datasets.

Dataset	Number of attributes	Number of training samples	Number of test samples
Adult1	123	1605	30956
Adult2	123	2265	30296
Web1	300	2477	47272

Table 5.5. Best classification performance on the ‘adult’ and ‘web’ test datasets for SVM, LSSVM and LSKMBC. Corresponding best parametric settings of each classifier is also reported.

Classifier	SVM (Gaussian)	SVM (Cubic)	SVM (Linear)	LSSVM (Gaussian)	LSKMBC (Gaussian)	LSKMBC (Gauss+Expo)	LSKMBC (Cauchy)
Adult1	84.22	80.15	84.26	82.1	84.14	84.14	84.28
Adult2	84.33	79.18	84.44	81.42	84.55	84.55	84.69
Web1	97.96	97.72	97.74	97.98	98.1	97.88	98.1

5.4.4. Performance of the λ -kNN Classifier

We report the 10-fold cross-validation results of the λ -kNN classifier as proposed in Section 3.2 for the datasets in Table 5.1. We computed the CV scores in the same way (random sampling) as described in Section 4.3.2. As a comparison we provide the corresponding scores of the standard k-nearest neighbor algorithm. We report the best scores obtained for both the algorithms and show the value of optimal λ for the λ -kNN and the value of optimal k for the standard kNN classifier. Table 5.6 summarizes the performance scores. We observe that λ -kNN is able to outperform the standard kNN algorithm in terms of mean accuracies on the ‘Bupa’, ‘Wdbc’, ‘Wpbc’, and ‘Ecoli’ datasets. The scores are same for the ‘Iris’ dataset whereas for the ‘Pima’ dataset, standard kNN performs better than the λ -kNN.

Table 5.6. 10-fold classification scores of kNN and λ -kNN over the datasets in Table 5.1. Scores summarize the mean accuracies and corresponding standard deviations.

Classifier	Pima	Bupa	Wdbc	Wpbc	Iris	Ecoli
λ -kNN	75.86 (± 4.12)	68.18 (± 8.46)	97.44 (± 2.13)	79.75 (± 7.11)	97.33 (± 4.44)	87.51 (± 5.40)
kNN	76.01 (± 3.71)	65.41 (± 7.08)	97.36 (± 2.13)	78.89 (± 6.93)	97.33 (± 4.44)	87.38 (± 4.68)

We also compare the performance of the λ -kNN classifier with the standard kNN classifier using the ‘adult’ and ‘web’ datasets in Table 5.7. Since the adult and web datasets contains sparse vectors of zeros and ones, we

used two different similarity measures to obtain the nearest neighbors. We used the standard Euclidian distance and also used normalized cosine similarity. Normalized cosine similarity is often used to measure the similarities between text documents. We observe that for the ‘adult1’ dataset, the standard kNN classifier performs better than the λ -kNN classifier when we compute neighbors based on the Euclidian distance; however, λ -kNN outperforms standard kNN classifier using cosine similarity. For the ‘adult2’ and ‘web1’ datasets, λ -kNN classifier performs better than the standard kNN with both Euclidian distance and cosine similarity.

Table 5.7. Classification performance of λ -kNN and standard kNN classifiers on ‘adult’ and ‘web’ datasets using Euclidian distance and cosine similarity.

Classifier	Adult1		Adult2		Web1	
	Euclidian	Cosine	Euclidian	Cosine	Euclidian	Cosine
λ -kNN	83.02	83.25	83.40	83.43	97.49	97.93
kNN	83.12	83.21	83.37	83.29	97.41	97.89

5.5. Summary and Conclusions

We presented an interpretation of the principle of parsimony in the context of pattern classification where we minimize the superfluity in the outcome of a classifier with a given margin. We described the meaning of the outcome in the context of different types of classifiers. We then provided an explanation of this principle from the bias-variance perspective where we have shown that the margin of the superfluity determines the boundary bias of the classifier. As the margin increases, the bias decreases and the variance increases. Similarly the bias increases with the decrease in the margin. In the limiting condition, as the margin goes to zero, it imposes infinite bias on the classifier. We also explained the connection of this principle with the regularization techniques using model complexity. The Lagrangian multiplier of the regularization functional increases with a decrease of the margin of superfluity in the outcome. This principle is different from the margin maximization principle used in the SVM classifier where margin denotes how far the samples are separated from the separating hyperplane. In our context, the margin acts as a threshold for the superfluity in the outcome of a classifier. Note that, we do not claim this principle as a new principle. This principle has been used in decision tree pruning where the leaf nodes of a tree are pruned till the impurity reaches a certain threshold. In context of decision tree growing, a leaf node is not further split if the impurity

reaches that threshold. The threshold in impurity is exactly the same as the margin in our principle. However, this mechanism has not been explicitly used to design other classifiers. We explicitly state this practice in the form of a principle and use this principle to design two different classifiers namely LSKMBC (which is kernel based classifier) and λ -kNN (which is based on k-nearest neighbor classifier). We show that for certain values of λ , LSKMBC is able to outperform the standard SVM on several datasets. We also observed that λ -kNN is able to outperform standard kNN classifier for several datasets.

We also discussed the relationship of the LSKMBC with certain other kernel based classifiers. We interpreted LSKMBC as a least square kernel machine with box constraint where it uses uniform priors instead of Gaussian priors. From the regularization perspective, LSKMBC has similarity with certain other kernel machines although the regularization functional is different in the LSKMBC. We mentioned that ν -SVM also uses a similar principle of regularizing the margin of separation between the classes (with an additional regularization factor) where the parameter ν controls the penalization due to separation of the classes in the loss functional. However, ν -SVM also uses the basic principle of margin maximization between the classes as used in SVM whereas the basic principle in LSKMBC is different. We mentioned that LSKMBC is flexible to accommodate non-Mercer kernels unlike the SVM and related kernel machines. We observed that use of centrally peaked long-tailed kernels can enhance the performance of the LSKMBC. In addition to Gaussian kernel, we used two other kernel functions namely, Cauchy kernel and a hybrid of Gaussian and exponential kernels. We observed that these long-tailed kernel functions provide better performance on several real-life datasets as compared to simple Gaussian kernel due to the long-range interaction between the B-vectors.

One disadvantage of the LSKMBC is that it produces relatively non-sparse solution as compared to SVM and other least square kernel machines using Hilbert kernels. This leads to relatively slower training processes for LSKMBC. In this respect, further investigation can be performed towards obtaining better variants of the LSKMBC. With respect to the run-time memory requirements, sequential minimal optimization of LSKMBC objective functional for larger datasets can also be investigated as a scope of future study.

References

- [1] Y. Tsyplkin, *Foundation of the Theory of Learning Systems*. Academic Press, New York (1973).
- [2] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification (2nd Edition)*. Wiley, New York (2001).
- [3] V. Vapnik, *The Nature of Statistical Learning Theory*. Springer-Verlag, New York (1995).
- [4] S. Amari, Theory of adaptive pattern classifiers, *IEEE Trans. EC-16*, 299–307 (1967).
- [5] J. H. Friedman, On bias, variance, 0/1-loss, and curse-of-dimensionality, *Data Mining and Knowledge Discovery*. **1**, 55–77 (1997).
- [6] R. Tibshirani. Bias, variance and prediction error for classification rules. Technical Report <http://www.utstat.toronto.edu/~tibs>, Dept. of Statistics, University of Toronto, Canada (1996).
- [7] S. Geman, E. Bienenstock, and R. Doursat, Neural networks and the bias/variance dilemma, *Neural Computation*. **4**, 1–58 (1992).
- [8] R. Kohavi and D. H. Wolpert. Bias plus variance decomposition for zero-one loss functions. In *Proceedings of the Thirteenth International Conference on Machine Learning (ICML96)* (1996).
- [9] T. M. Cover. Learning in pattern recognition. In ed. S. Watanabe, *Methodologies of Pattern Recognition*, 111–132. Academic Press, New York (1969).
- [10] M. Stone, Cross-validatory choice and assessment of statistical predictions, *Journal of the Royal Statistical Society B*. **36**, 111–147 (1974).
- [11] M. Stone, Asymptotics for and against cross-validation, *Biometrika*. **64**, 29–35 (1977).
- [12] A. K. Jain, R. C. Dubes, and C. Chen, Bootstrap techniques for error estimation, *IEEE Trans. Pattern Analysis and Machine Intelligence*. **PAMI-9**(5), 628–633 (1987).
- [13] B. Efron, Estimating the error rate of a prediction rule : Improvement on cross-validation, *Journal of the American Statistical Association*. **78**(382), 316–330 (1983).
- [14] J. Shao, Linear model selection via cross-validation, *Journal of the American Statistical Association*. **88**(422), 486–494 (1993).
- [15] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence (IJCAI-95)* (1995).
- [16] P. Zhang, On the distributional properties of model selection criteria, *Journal of the American Statistical Association*. **87**(419), 732–737 (1992).
- [17] P. Smyth and D. Wolpert, An evaluation of linearly combining density estimators via stacking, *Machine Learning*. **36**, 59–83 (1999).
- [18] L. Breiman, Bagging predictors, *Machine Learning*. **26**, 123–140 (1996).
- [19] L. Breiman, Stacked regressions, *Machine Learning*. **24**, 49–64 (1996).
- [20] L. Breiman, Arcing classifiers, *The Annals of Statistics*. **26**, 801–824 (1998).
- [21] R. E. Schapire and Y. Singer, Improved boosting algorithms using confidence-rated predictions, *Machine Learning*. **37**, 297–336 (1999).

- [22] A. N. Tikhonov and V. Y. Arsenin, *Solutions of Ill-posed Problems*. W. H. Winston, Washington, D. C. (1977).
- [23] J. Rissanen, Modeling by shortest path description, *Automatica*. **14**, 465–471 (1978).
- [24] V. Vapnik and A. Chervonenkis, *Theory of Pattern Recognition*. Nauka, Moskow, Russian edition (1974).
- [25] H. Akaike, A Bayesian analysis of the minimum AIC procedure, *Annals of the Institute of Statistical Mathematics*. **30A**, 9–14 (1978).
- [26] G. Schwarz, Estimating the dimension of a model, *Annals of Statistics*. **6**, 461–464 (1978).
- [27] N. Murata, S. Hoshizawa, and S.-I. Amari. Learning curves, model selection and complexity in neural networks. In eds. S. J. Hanson, J. D. Cowan, and C. L. Giles, *Advances in Neural Information Processing Systems (NIPS92)*, **5**, 607–614. MIT Press, Cambridge, MA (1993).
- [28] P. E. Hart, The condensed nearest neighbor rule, *IEEE Trans. Information Theory*. **IT-14**, 515–516 (1968).
- [29] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Chapman & Hall, New York (1983).
- [30] J. R. Quinlan, *Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA, USA (1993).
- [31] D. W. J. Hosmer and S. Lemeshow, *Applied logistic regression* (2nd ed.). John Wiley, USA (2000).
- [32] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Prentice Hall, Upper Saddle River, NJ (1999).
- [33] M. I. Jordan and R. A. Jacobs, Hierarchical mixtures of experts and the EM algorithm, *Neural Computation*. **6**, 181–214 (1994).
- [34] C. M. Bishop, *Pattern recognition and machine learning*. Springer, UK (2006).
- [35] Y. Le Borgne. Bias-variance trade-off characterization in a classification problem: What differences with regression? Technical Report 534, Machine Learning Group, Univ. Libre de Bruxelles, Belgium (2005).
- [36] R. Rifkin and A. Klautau, In defense of one-vs-all classification, *Journal of Machine Learning Research*. **5**, 101–141 (2004).
- [37] N. Cristianini and J. Shawe-Taylor. Support vector machines. URL <http://www.support-vector.net/software.html> (2000).
- [38] T. D. Bie, Lanckriet, G. R. G., and N. Cristianini. Convex tuning of the soft margin parameter. Technical Report UCB/CSD-03-1289, University of California, Computer Science Division (EECS), Berkeley, California, USA (2003).
- [39] J. Basak. A least square kernel machine with box constraints. In *Proceedings of the international conference on Pattern Recognition(ICPR 08), Tampa, Florida, USA* (2008).
- [40] J. Basak, A least square kernel machine with box constraints, *Journal of Pattern Recognition Research*. **5**, 38–51 (2010).
- [41] P.-H. Chen, C.-J. Lin, and B. Schölkopf, A tutorial on nu-support vector machines, *Applied Stochastic Models in Business and Industry*. **21**, 111–136 (2005).

- [42] D. W. Marquardt, Generalized inverses, ridge regression, biased linear estimation, and nonlinear estimation, *Technometrics*. **12**, 591–612 (1970).
- [43] C. Saunders, A. Gammerman, and V. Vovk. Ridge regression learning algorithm in dual variables. In *Proceedings of the 15th international conference on machine learning (ICML98)* (1998).
- [44] R. Rifkin, G. Yeo, and T. Poggio. Regularized least square classification. In eds. J. Suykens, G. Horvath, S. Basu, C. Micchelli, and J. Vandewalle, *Advances in learning theory: Methods, model and applications, NATO science series III: Computer and system sciences*, **190**, 131–153. IOS Press, Amsterdam (2003).
- [45] J. A. K. Suykens and J. Vandewalle, Least square support vector machine classifiers, *Neural Processing Letters*. **9**, 293–300 (1999).
- [46] T. Van Gestel, J. Suykens, B. Baesens, S. Viaene, J. Vanthienen, G. Dedenne, and B. De Moor, Benchmarking least squares support vector machine classifiers, *Machine Learning*. **54**, 5–32 (2004).
- [47] R. Tibshirani, Regression shrinkage and selection via LASSO, *Journal Royal Statistical Society, Series B*. **58**, 267–288 (1998).
- [48] M. Osborne, B. Presnell, and B. Turlach, On the LASSO and its dual, *Journal Comput. Graphical Statist.* **9**, 319–337 (2000).
- [49] V. Roth, The generalized LASSO, *IEEE Trans. Neural Networks*. **15**, 16–28 (2004).
- [50] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. Technical Report NC-TR-98-030, NeuroCOLT, Royal Holloway College, University of London, UK (1998).
- [51] M. Tipping, Sparse Bayesian learning and the relevance vector machine, *Journal Machine Learning Research*. **1**, 211–244 (2001).
- [52] C. Blake and C. Merz. UCI repository of machine learning databases. URL [http://www.ics.uci.edu/\\$\sim\\$mlearn/MLRepository.html](http://www.ics.uci.edu/\simmlearn/MLRepository.html) (1998).
- [53] K. Nakai and M. Kanehisa, Expert system for predicting protein localization sites in gram-negative bacteria, *PROTEINS: Structure, Function, and Genetics*. **11**, 95–110 (1991).
- [54] P. Horton and K. Nakai. A probabilistic classification system for predicting the cellular localization sites of proteins. In *Intelligent Systems in Molecular Biology*, pp. 109–115, St. Louis, USA (1996).
- [55] B. A. Olshausen and D. J. Field, Emergence of simple-cell receptive field properties by learning a sparse code for natural images, *Nature*. **381**, 607–609 (1996).
- [56] S. Canu, Y. Grandvalet, and A. Rakotomamonjy. Svm and kernel methods matlab toolbox. Perception Systemes et Information, INSA de Rouen, Rouen, France. URL <http://asi.insa-rouen.fr/%7Earakotom/toolbox/index> (2003).
- [57] K. Pelckmans, J. A. K. Suykens, T. Van Gestel, J. De Brabanter, L. Lukas, B. Hamers, B. De Moor, and J. Vandewalle. Ls-svmlab toolbox users guide. Technical Report 02-145 (<http://www.esat.kuleuven.ac.be/sista/lssvmlab/>), Department of Electrical Engineering, ESAT-SCD-SISTA, Katholieke Universiteit Leuven, Belgium (2003).

- [58] T. G. Dietterich, Approximate statistical test for comparing supervised classification learning algorithms, *Neural Computation*. **10**, 1895–1923 (1998).
- [59] J. C. Platt. Sequential minimal optimization : A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, Microsoft Research, USA (1998).

Chapter 6

Robust Learning of Classifiers in the Presence of Label Noise

P. S. Sastry¹ and Naresh Manwani²

¹*Dept. Electrical Engineering
Indian Institute of Science, Bangalore, India
sastry@ee.iisc.ernet.in*

²*Microsoft India (R&D) Pvt. Ltd.
India Development Center, Bangalore, India
nareshmanwani@gmail.com*

Algorithms for learning pattern classifiers should, in general, be able to handle noisy training data. An important case is when the class labels in the training data are noise-corrupted. In many applications today, training data is to be obtained through crowd sourcing which naturally introduces such label noise. There are other reasons also for training data being corrupted by label noise. Many of the standard classifier learning strategies are seen to be sensitive to noisy labels in training data. Hence one needs special techniques to handle label noise. In this chapter we present an overview of some of the methods that are proposed for dealing with label noise. We elaborate more on some of the recent methods for robust learning of classifiers.

6.1. Introduction

Pattern classification is a basic problem in Machine Learning. One learns the classifier using a training set of labeled examples. In most situations of learning a classifier, one has to contend with noisy examples. We say that the training examples are corrupted with label noise when the class labels of examples as provided in the training set may not be ‘correct’. Such noise in training data can come through many sources and it is often unavoidable. Hence it is always desirable to have classifier design strategies that are robust to noise in training data.

Most of the standard classifier learning algorithms (*e.g.*, Perceptron, support vector machine, Adaboost, etc.) perform well only under noise-free training data. When there is label noise, the classifiers learnt by them are severely affected by the noisy points. (See, for example, [1]).

Over the years, many techniques have been developed for dealing with noisy labels in training data [2]. To learn classifiers in presence of label noise, one can pre-process the data to filter out the noisy examples (or correct the labels on them) and then learn a classifier using standard approaches. Alternatively one can devise classifier learning methods which are robust to label noise.

In this chapter, we provide an overview of various techniques proposed for learning under label noise. We discuss various sources of label noise and motivate the importance of developing noise tolerant classifier learning techniques. We present a formalism for label noise and discuss different types of label noise. We then provide an overview of many of the algorithms proposed for learning under label noise. We also discuss, in some detail, some recent developments in noise-tolerant risk minimization methods which are quite effective for learning under label noise.

6.2. Label Noise

Supervised learning of classifiers is a basic problem in machine learning. The standard approach here involves learning the classifier from a training set of examples with known class labels. In most applications, the training data that one can get may be corrupted with noise. When the class labels for examples as given in the training data may be incorrect, we say that the training set is corrupted with label noise. Learning a classifier using training data with label noise may be called *imperfectly supervised* learning [2].

There are many sources of label noise. A major reason for label noise is that the training data is often obtained through manual labeling. There are many reasons for labeling errors in such a situation. For example, in a medical diagnosis application, labelling of a pattern (consisting of some biomedical images or test results) by different experts may give rise to label noise because expert opinions can differ. Also, sometimes human experts are asked to label based on insufficient evidence and this would also give rise to subjective errors in labeling. There are also many applications (*e.g.*, classifying images or documents) where training data is obtained through mechanisms such as crowd sourcing. These also give rise to label noise due to the variability in the quality of different labelers. This subjectivity

may also arise when the information used to label an object is different from the information available to the learning algorithm [3]. For example, when labeling an image, the analyst uses the visual information. But the learning algorithm is given access to numeric values of the feature vector corresponding to that image. Another reason for label noise here is that in many cases the labeler has to choose between predefined categories and the different categories may be imprecisely defined. Human errors in data entry also give rise to label noise.

In the standard statistical framework of pattern recognition, a major source of noise in training data is overlapping class conditional densities [4, 5]. If the class conditional densities overlap, then same feature vector can come from different classes with different probabilities. This can also be viewed as a form of label noise because here the label that an *optimal* classifier assigns to a feature vector may not be same as the label given for that feature vector in the training data.

There can also be applications where the process for generating training examples may itself be noisy. We detail an interesting example of this which is from [6]. High energy neutrons are often detected through a scintillation detector whose output is a voltage waveform which is converted into a feature vector by sampling it. The detector is also sensitive to gamma rays and hence gamma rays also elicit some voltage pulse from the detector. To be able to properly characterize nuclear materials etc, it is important to have a classifier that can discriminate between the voltage pulses that are recorded due to high energy neutrons and those that are recorded due to gamma rays. Since the fission events that generate neutrons also generate gamma rays, and gamma rays and neutrons can always be there in the background, one cannot generate example outputs of detectors that are purely of one class. If we are generating neutrons and recording the outputs, some recordings we get may actually be due to the gamma rays that are also generated. Thus, the examples would have label noise. Since one can get strong gamma ray sources which do not generate neutrons, the noise here may be such that examples of one class may have higher label noise than those of the other class.

As we have noted earlier, most standard classifier learning techniques get adversely affected by label noise [1, 7, 8]. A related problem is the difficulty in validation. Most learning algorithms have parameters to be fixed by the user and often these are tuned based on the performance of the learnt classifier on a subset of data called validation data. When data is corrupted with label noise, the inferred results from validation would be

erroneous. Thus, we may end up using wrong parameter values also in the learning algorithm which adds to the difficulty in learning classifiers under label noise.

Since label noise is unavoidable in many applications, we need learning strategies that are robust to such noise. In this chapter we provide an overview of the many methods that are suggested for learning in the presence of label noise. Before discussing the different methods for tackling label noise, we will first present a simple formalism to specify label noise and define different types of label noise.

6.2.1. Types of Label Noise

Let $\mathcal{X} \subset \mathbb{R}^d$ be the feature space and let \mathcal{Y} be the set of class labels. Each training example would be a feature vector along with a class label. Thus examples are drawn from $\mathcal{X} \times \mathcal{Y}$. To formalize the issues involved in learning with label noise we think of an ideal noise-free training data from which the real training data (corrupted by label noise) is obtained. The noise-free data is unobservable in the sense that we have no access to it. But ideally, we want techniques that learn a classifier from the noisy data so that the learnt classifier has performance close to that of a classifier which can be learnt from the ideal noise-free data.

Let $S = \{(\mathbf{x}_1, y_{\mathbf{x}_1}), (\mathbf{x}_2, y_{\mathbf{x}_2}), \dots, (\mathbf{x}_N, y_{\mathbf{x}_N})\} \in (\mathcal{X} \times \mathcal{Y})^N$ be the (unobservable) noise free data, drawn *iid* according to a fixed but unknown distribution \mathcal{D} over $\mathcal{X} \times \mathcal{Y}$. The noisy training data given to learner is $S_\eta = \{(\mathbf{x}_i, \hat{y}_{\mathbf{x}_i}), i = 1, \dots, N\}$, where $\hat{y}_{\mathbf{x}_i} = y_{\mathbf{x}_i}$ with probability $(1 - \eta_{\mathbf{x}_i})$ and with the remaining probability $(\eta_{\mathbf{x}_i})$, $\hat{y}_{\mathbf{x}_i}$ would be one of the other labels (with, *e.g.*, all other labels being equally probable). *Note that our notation allows that the probability that the label of an example is incorrect may be a function of the feature vector of that example.* As a notation, for a feature vector \mathbf{x} , its correct label (that is, label under distribution \mathcal{D}) is denoted as $y_{\mathbf{x}}$ while the noise corrupted label is denoted by $\hat{y}_{\mathbf{x}}$. As noted above, the joint distribution of \mathbf{x} and $y_{\mathbf{x}}$ is given by \mathcal{D} . We use \mathcal{D}_η to denote the joint probability distribution of \mathbf{x} and $\hat{y}_{\mathbf{x}}$. The $\eta_{\mathbf{x}}$ is often referred to as the noise rate.

Depending on the assumptions we make on the noise rate (or the probability with which the label is corrupted), we can broadly classify label noise into the following types.

- (1) **Uniform Label Noise:** Here we assume $\eta_x = \eta$, $\forall x$. Thus the noise rate is same for each point irrespective of its class and location in feature space. This is the simplest case of label noise.
- (2) **Class Conditional Label Noise:** In this type of noise, noise rate for each class is different. But inside each class the noise rate is constant. Thus $\eta_x = \eta_C$, $\forall x \in C$.
- (3) **Constant Partition Label Noise:** Here, feature space is divided into different partitions and each partition has different noise rate [9]. But inside a partition the noise rate is constant. Class conditional noise is a special case of constant partition noise where each partition corresponds to one class region.
- (4) **Monotonic Label Noise:** Here we allow η_x to vary with x but with some restrictions. Under monotonic classification noise, the probability of mislabeling an example monotonically decreases with the distance of the example from the class boundary [10]. This essentially means that feature vectors closer to class boundary have higher probability of being mislabelled compared to feature vectors far from the class boundary.
- (5) **Non-uniform Label Noise:** This is the most general case of noise where η_x is a general function of x . Under this noise, each point can have a different noise rate. All the earlier types of label noises are special cases of non-uniform noise.

In the above, we have only considered the label noise. Another general type of noise considered in the literature is the so called attribute noise. Here the attributes or the feature vector components may have their values noise corrupted. Such a noise can, in general, alter the observed statistics of attribute values. When a pattern sitting deep inside the class regions is perturbed, it is unlikely that the perturbed pattern will shift to other class regions. But when the pattern close to the classification boundary is perturbed, it is likely that it will get shifted to other class regions. So, depending on how close the actual pattern is from the classification boundary, its perturbed version may appear as a point of other class. Because of this reason, attribute noise can be thought of as subsumed in nonuniform label noise model. On the other hand, non-uniform noise appears more general than any attribute noise because it can corrupt labels of patterns sitting deep inside the class regions also. In this article we consider only the label noise.

6.2.2. Noise Tolerant Approaches: A Broad Classification

In this section we provide a broad categorization of the different strategies for dealing with label noise and present a brief summary of each of these approaches. In the next few sections we discuss different methods proposed under each of these broad strategies.

- (1) **Noise Cleaning (Restoring Correct Labels):** One approach to handle the noisy data is to try and restore the correct labels for all training examples. Once all the labels are cleaned, the training data essentially becomes noise-free and we can learn a classifier using any standard technique. Since we do not know which are the mislabeled points, each point has to go through the process of noise cleaning. In the process of cleaning we may alter labels on points which have correct labels and we may leave wrong labels on noisy points. The final performance depends on the ability of the algorithm to minimize such errors.
- (2) **Eliminating Noisy Points:** Another approach for dealing with noisy data is to detect and eliminate training examples with wrong labels. In these approaches, the quality of class label of each point is assessed through some method (*e.g.*, an auxiliary classification system) and this is used to filter the noisy points out of the training data. Mislabeled points may be similar to outliers and hence many techniques for outlier detection are useful here. Once the noisy points are removed, we can learn a classifier on the remaining noise-free data. The success of the approach depends on whether we can avoid removing clean points and retaining noisy points.

Even if all the noisy points are accurately identified and removed from the training dataset, we may still face problems in terms of learning a classifier with reduced number of training examples. The classifier built on the reduced size training data may have poor generalization abilities. Since mislabeled points can be viewed as outliers, deletion of these points can lead to reduced bias of the classifier [11]. On the other hand, because of the reduced size of training data, the variance may increase [11].

The noise cleaning approach described earlier may seem better because it allows one to use all the training examples unlike elimination of noisy examples where the final training set size reduces. However, in practice, it is difficult to correctly restore the clean labels for *all* examples. On the

other hand, by being a little conservative we may be able to ensure that almost all noisy examples (along with, may be, some good examples) are removed. Hence, approaches based on eliminating noisy examples are much more popular than the approach of restoring correct labels to all examples.

- (3) **Mitigating the effect of Mislabeled Points:** In these set of approaches instead of removing the noisy points or changing the labels on examples, the learning algorithm is designed so as to reduce the effects of label noise on the final learnt classifier. These methods do not perform any data preprocessing. Instead the learning algorithm may be modified to reduce the sensitivity of final learnt classifier to label noise. Most of these methods are based on heuristics about how label noise affects the learnt classifier.
- (4) **Noise Tolerant Algorithms:** All the above discussed approaches are designed to treat the noisy data differently than the clean data. In contrast, there are algorithms which are designed so that the classifier they can learn with noisy data would be close to the classifier they would have learnt if they had been given the noise-free data. In this sense they are unaffected by noise. The underlying idea in these methods is to exploit some statistical properties that are robust to label noise. We call these as *noise tolerant approaches*.

Learning under label noise is a hard problem. There are many results to show that standard learning strategies are not robust to label noise. For example, it is shown through counter examples in [8] that risk minimization under hinge loss, exponential loss and logistic loss is not tolerant to even uniform noise. Similarly, it is known that boosting techniques are severely affected by label noise [7]. However, there are also some positive results on being robust to label noise. For example, as we see later on, it is possible to have risk minimization strategies that are provably robust to label noise.

There are also some correctness results under the Probably Approximately Correct (PAC) learning framework for learning under label noise. It is possible to learn linear threshold functions in presence of uniform label noise under some data distributions [12]. It is possible to learn linear classifiers in the presence of uniform and class-conditional noise [13, 14] (assuming that the best linear classifier has 100% accuracy under noise-free distribution).

In our overview here, we will discuss many algorithms that are mainly heuristic but have been found useful in many applications. We would also

be discussing algorithms that are provably noise-tolerant to different types of noise. Towards the end of the chapter, we discuss a generic method for learning under label noise (based on risk minimization) which is provably noise-tolerant.

6.3. Noise Cleaning

As explained earlier, noise cleaning means restoring clean labels for every training example. Once the data has been cleaned, we can learn a classifier using any standard method. Fine *et al.* [15] suggest a simple idea for noise cleaning. For each data point we find k nearest neighbors and then assign to the data point the label of majority class among those k -nearest neighbors. The motivation is that, most of the time, for any point in the feature space many points in its neighbourhood would be in the same class. Since not all nearest neighbours of a point are likely to have corrupted labels, the majority class among the nearest neighbours would be a reasonable guess for the class label of any point. This heuristic reasoning may not work for points near the class boundaries. However, if we assume some minimum margin between any pair of classes, then the above heuristic is again reasonable. The nearest neighbours need not be defined based only on Euclidean distance. For example, for classification problems with Boolean features the definition of neighborhood can be different. The problem is formulated as a dual learning problem in [15]. Given a prior probability distribution on the hypothesis space and assuming some minimum margin between the two classes, one can guarantee the ability of the algorithm in cleaning noisy labels [15]. In practice, this method works well when noise is uniform and noise rate is low. In such a case (with the assumption of a lower bound on margin), the only noisy labels that may be left are likely to be only near class boundaries.

6.4. Eliminating Noisy Examples

This approach aims at identifying training examples with wrong labels and then eliminating them from the training set. A good classifier can then be learnt using the remaining training examples whose labels (hopefully) are all correct.

Identifying mislabeled examples is, in general, a hard problem. If we have a good classifier then, we can easily identify examples with wrong labels. However, we cannot learn a good classifier when examples have

label noise. Some approaches try and identify noisy examples by using some learnt classifiers.

Since the neighbours of a mislabeled example are all likely to be examples of a different class, one can view mislabeled examples as outliers. So, outlier detection techniques can also be adopted for this problem.

In general most of these approaches are heuristic and each method works well in some situations.

In this section we briefly outline a representative subset of the algorithms that are proposed for identifying noisy examples.

6.4.1. *Using disagreement of multiple classifiers*

Since the noisy examples have wrong labels, they are the ones which are tough to learn for any classifier. If we learn multiple classifiers on subsamples of training data then each classifier will be affected by the noisy examples differently. And most likely all the classifiers will misclassify the noisy examples. This idea can be used for detecting noisy examples which is sometimes called *classification filtering*. This idea has been used in [3], where the multiple training sets are created through n -fold cross validation. For each of the n ways of splitting the data into training and validation sets, m classifiers are trained using different algorithms. Thus, for each training example, there are m classifiers which are learnt without using that example and thus we can get m classification decisions for each training example. Then we declare a point to be noisy if a majority of the classifiers misclassify it. In some versions of *classification filtering*, a point is declared noisy only if all the m classifiers misclassify it and this reduces the chance of false positives (at the expense of retaining some noisy examples). There are variations on this basic method depending on the type of classifier ensemble one uses and the specific voting method used to decide whether an example is noisy. Such ideas have been found useful in many applications. For example, multiple classifiers learnt using subsamples from the training set are used to detect and eliminate noisy examples in the context of learning object categories such as human faces [16].

6.4.2. *Noise elimination using decision tree pruning*

A method for making decision tree learning robust to outliers is proposed in [17]. In the presence of outliers the size of learnt decision trees increases as more decision nodes are required to fit the outliers. To overcome this different pruning schemes have been proposed in literature. In any pruning

process, some decision nodes from the bottom of the tree are converted to leaf nodes such that the accuracy of the tree does not change significantly. Pruning has the same effect as that of ignoring all training examples which come to the pruned decision node and are misclassified by that node. Which means that the pruning process assumes that those patterns were locally uninformative. In [17] this idea is extended by assuming that these patterns are globally uninformative and can be thought of as outliers or examples with noisy labels. In their method, these patterns are removed from the training data and the whole decision tree is relearnt. Experimentally it is observed that this approach, on the average, increases classification accuracy and decreases tree size.

6.4.3. VC theory based approach

Another method for data cleaning was proposed in [18] using VC theory. They derived the following upper bound, G , on the error on the validation set.

$$G = E_{train}(m) + \beta \frac{d(\ln \frac{2(p-m)}{d} + 1) + m(\ln \frac{p}{m} + 1)}{p-m} \quad (6.1)$$

Here d is the VC dimension, p is the number of points in the original training set (before cleaning) and m is the number of points removed from the training set before training the current classifier. $E_{train}(m)$ is the training error rate when m suspicious points are removed from training set. The idea is that we can decide how many candidate outliers to remove by trying to minimize the above upper bound.

6.4.4. Nearest neighbor and majority voting

Another outlier detection technique is proposed in [19] for identifying examples with noisy labels. According to this, if any point is closer to the centroid of some class other than its own class, then that point is a candidate for being an outlier. For each such candidate outlier, k nearest neighbors are found and if the majority class differs from the label of the point then that point is declared as a noisy point. Although this approach seems appealing for detecting points with wrong class labels, its effectiveness is very much dependent on the shape of different class regions. The approach is reasonable when all class regions are nearly spherical.

6.4.5. Maximum margin approach

Suppose we have a training set of n examples and suppose we know that k of the points are noisy. Then if we remove those k noisy points, the remaining set of training examples will be non-noisy. Now the task is to identify those $(n - k)$ non-noisy points. One idea is that the hypothesis learnt on non-noisy points will have maximum possible margin (of separation between the classes). This idea has been used in [20]. There are C_{n-k}^n different ways of choosing a subsample from the training set. A naive approach is learning large margin classifiers on all possible subsamples and choosing the classifier giving maximum margin. An efficient search approach is proposed in [21] using core sets to find the subset of training data of size $(n - k)$ which gives maximum margin classifier.

There is no guarantee that removing the noisy points gives rise to a maximum margin classifier. The maximum margin classifier being the desirable one is only a heuristic. It is reasonable to infer that this approach is good when most of the data points with wrong labels are near the classification boundary. Because removing the points near the classification boundary will lead to a new hypothesis having greater margin.

6.4.6. Modifying the objective function of support vector machine (SVM) classifier

The Support Vector Machine (SVM) classifier is not very tolerant to label noise [1, 8]. In the standard SVM formulation [22] the hyperplane parameters are obtained as weighted sum of some of the training examples (which are the so called support vectors). Normally outliers would become support vectors and their weights attain the upper bound. This is one reason why the learnt hyperplane is sensitive to outliers. To take care of outliers, SVM formulation can be modified by introducing one variable for each example that decides whether that example is to be viewed as outliers [23] and make these variables also as part of the optimization variables. Now, the optimization problem becomes

$$\min_{\mathbf{w}} \min_{0 \leq \eta_i \leq 1} \frac{1}{2} \|\mathbf{w}\|^2 + \sum_i^n \{\eta_i \max(0, 1 - y_i \mathbf{w}^T \mathbf{x}_i) + (1 - \eta_i)\}$$

Essentially, we can choose to keep η_i close to 1 and then suffer penalty on misclassifying the i^{th} example or choose to keep η_i close to zero and then suffer a fixed penalty (irrespective of whether or not the i^{th} example is classified correctly). The idea is that the optimization process would

ensure η_i close to 1 if \mathbf{x}_i has correct label and is close to zero otherwise. Solving this problem gives a robust classifier. This optimization problem becomes a convex quadratic program in \mathbf{w} when η_i are fixed and becomes a linear program in η_i . Now since this formulation is not convex with respect to \mathbf{w} and η_i together, an alternating optimization scheme is used. It has been observed that for a fixed \mathbf{w} if we optimize with respect to η_i only, the solution appears to be binary valued [23]. Which means that η_i appears to be 0 for all misclassified points and 1 for the rest. After solving the problem, the values of η_i can be used to tag the mislabeled points. This formulation can also be modified to put a constraint to specify a maximum limit on the fraction of points that can be thought of as outliers [23].

6.4.7. Noise elimination using modified boosting approach

A modified boosting algorithm is proposed in [24] for identifying the outliers and removing them. In boosting based approaches, multiple training sets are obtained by sampling from the original training set using weights assigned to each example. The weights are updated after each iteration so that points misclassified by previous classifiers get their weights increased. Since points with wrong class label are difficult to learn for most classifiers these noisy points in the training set keep getting higher and higher weights. In [24] an upper bound is set on the weights that any sample point can have. If the weight of any point exceeds this bound, then that point is considered to be an outlier and its weight is set to zero. Thus, after some iterations, most of the noisy points are effectively removed from the training data. Then a new classifier can be learnt without any of the noisy points.

6.4.8. Noise cleaning on streaming data

So far all the methods that we mentioned assume that the data is static in the sense that all the data is stored and is available to the learning algorithm. When we have to learn classifiers using streaming data, we may not be able to store all data. Further, in streaming data applications, the required classifier may slowly change with time. This is called concept drift. This makes cleaning stream data a difficult problem. Stream data cleaning problem is treated by dividing the data into chunks. Data cleaning can be done in two ways: locally and globally.

In local cleaning, data in only a single chunk is used. This single chunk data is divided in two parts. An ensemble of classifiers is learnt on the first

part and is used to clean data in the other part (*e.g.*, by using majority voting). The problem with local cleaning is that each chunk contains only limited number of examples and the classifiers trained on one part in a single chunk may not be able to identify noisy points well. In a global cleaning method, data chunk S_i is cleaned using classifiers C_{i-k}, \dots, C_{i-1} learnt on previous k data chunks S_{i-k}, \dots, S_{i-1} [25]. The main disadvantage of this approach is that, in the presence of concept drift, classifiers learnt on previous data chunks may have no relevance in classifying the data in the current chunk. A greedy approach for cleaning streaming data is proposed in [25]. This is a mix of both local and global cleaning approaches.

6.5. Restricting The Effect of Noisy Points

In this section we explain a few approaches that try to modify a standard learning algorithm so that it becomes robust to label noise in training data. Here, one is not interested in explicitly detecting which training examples have wrong labels. The objective is to modify a learning algorithm so that the training data with wrong labels would not influence the final learnt classifier too much. In this sense, the modification of SVM that we considered in Sec.6.4.6 can be thought of as an example of this approach also.

6.5.1. Improving boosting algorithms

As mentioned earlier, in boosting approaches, successive classifiers are learnt using resampled training data where weights assigned to examples are used in sampling. Essentially, the weights are increased for data points which are misclassified by the currently learnt classifiers so that the next classifier pays more attention to those points. A typical weight update is given below.

$$W_i^T = \frac{e^{-H_{T-1}(\mathbf{x}_i)y_i}}{\sum_{i=1}^N e^{-H_{T-1}(\mathbf{x}_i)y_i}} \quad (6.2)$$

Here, W_i^T is the weight assigned to i^{th} example (\mathbf{x}_i, y_i) at T^{th} iteration and $H_{T-1}(\mathbf{x})$ is the classification decision based on all the base classifiers learnt in iterations 1 to $T - 1$. Whenever, this classification decision differs from y_i , the weight is increased. (Here we are considering 2-class case with +1 and -1 as the two class labels). Since the noisy points are difficult to learn, most base classifiers would misclassify them and thus the weight of noisy points keeps increasing with iterations. Therefore, the noisy points dominate the training set and hence the algorithm starts overfitting noisy

points thus resulting in poor generalization. Adaboost, a popular boosting algorithm, is well known for its overfitting nature on noisy points.

A modification of the boosting algorithm is suggested in [26], by using what may be called input dependent regularizer. What is proposed is a new form for the function $H_{T-1}(\mathbf{x})$ in which the coefficients of each base classifiers are made input dependent in the following way.

$$H_{T-1}(\mathbf{x}) = \sum_{i=1}^{T-1} \alpha_t e^{-|\beta H_{t-1}(\mathbf{x})|} h_t(\mathbf{x})$$

where $h_t(\mathbf{x})$ is the decision of the base classifier, learnt in the t^{th} iteration, on \mathbf{x} . The decision $h_t(\mathbf{x})$ is considered seriously only when the previous classifiers $H_{t-1}(\mathbf{x})$ are not confident about their decision. Furthermore introduction of this new factor makes the base classifier $h_t(\mathbf{x})$ to be consistent between the training phase and the prediction phase. Which means, for prediction $h_t(\mathbf{x})$ is used particularly on those type of input points on which it has been trained. The factor β here is used to decide to what extent opinion of $H_{T-1}(\mathbf{x})$ is taken seriously. In this case, the weight update is done using following equation

$$W_i^T = \frac{e^{-H_{T-1}(\mathbf{x}_i)y_i - |\beta H_{t-1}(\mathbf{x}_i)|}}{\sum_{i=1}^N e^{-H_{T-1}(\mathbf{x}_i)y_i - |\beta H_{t-1}(\mathbf{x}_i)|}}$$

It is shown that under the condition that the weighting coefficients α_t are bounded by some fixed constant α_{max} , the weight of each data pattern will grow at most polynomially with T . As a result the problem of overemphasizing the noisy points in Adaboost is controlled significantly [26].

6.5.2. Improving perceptron algorithm against outliers

Perceptron [27] is a classical online algorithm to learn linear classifiers in a 2-class problem. It tries to find a hyperplane that separates the examples of the two classes. Essentially, it classifies the next example using $\text{sign}(\mathbf{w}^T \mathbf{x} - \theta)$, where \mathbf{w} is the normal to the hyperplane and θ is the bias. If the classification is incorrect then the algorithm updates the \mathbf{w} and θ . Thus each time an example is misclassified, that example is used in updating the hyperplane. It is known that when the data is linearly separable, the perceptron algorithm will converge in finitely many iterations and find a separating hyperplane. However, if the data is not linearly separable, the corrections to the hyperplane would never end and the algorithm goes into an infinite loop. As shown in [28], when the data is not linearly separable,

there would be a subset of examples (which may be viewed as the being responsible for nonseparability) which keep getting misclassified and keep participating in updating the hyperplane. When there are mislabeled points in the data, the noisy data may be linearly nonseparable (even though the noise-free data is linearly separable). Thus the perceptron algorithm can get severely affected when there is label noise.

There have been many modifications of the perceptron algorithm suggested to handle label noise. (See [29] for a survey of noise tolerant variants of perceptron algorithm). We briefly describe below some of these modifications to the perceptron algorithm for learning in the presence of label noise.

- **Perceptron with fixed negative margin:** The modification here is that even if an example is misclassified (that is, it falls on the wrong side of the current hyperplane) it does not result in updating of the hyperplane if it is within some distance from the hyperplane. The main idea is to allow some margin around the hyperplane as the margin for noise. This margin can be fixed beforehand or can be adaptively changed during the learning process. This method is quite effective when training examples with wrong labels are close to the class boundary.
- **λ -trick:** Since the noisy points are responsible for nonseparability of data, we expect them to participate often in updating the hyperplane. Hence, if a point is participating frequently in the update of the hyperplane, then it can be considered as noisy point and it is allowed more negative margin, so that, algorithm stops updating the hypothesis whenever it is misclassified again. To take care of noisy points deep inside the class regions, an adaptive strategy can be used where the negative margin can be made dependent on the input pattern [29, 30]. In this modified perceptron algorithm, an example \mathbf{x}_j is classified based on $\text{sign}(\mathbf{w}^T \mathbf{x}_j + I_j y_j \lambda \|\mathbf{x}_j\|^2 - \theta)$, where I_j is an indicator variable such that $I_j = 1$ if \mathbf{x}_j was previously classified incorrectly and it is zero otherwise. Thus, during training, if \mathbf{x}_j is misclassified at some time then in future iterations \mathbf{x}_j is classified with respect to $\mathbf{w} + y_j \lambda \mathbf{x}_j$. A high value of λ makes the term $y_j \lambda \|\mathbf{x}_j\|^2$ dominate and makes sure that \mathbf{x}_j does not participate in further update. Effectively \mathbf{x}_j is ignored.
- **α -bound:** Another way of mitigating the effect of noisy points is to restrict the number of times an example is allowed to participate in updating the hyperplane. We put an upper bound, α , on the number of times any example is allowed to update the hyperplane. Once an

example is misclassified this many times, that example is effectively eliminated from the training data.

6.5.3. Instance weighting with label confidence

Effect of outliers can also be restricted by weighting instances with the confidences on their current labels. Let $P(l|\mathbf{x})$ be the confidence in the label l for point \mathbf{x} . If we find these confidences for all training points, a new classifier can be built by giving low weight to the points with inconsistent labels. Approaches based on mixture models have been used to find $P(l|\mathbf{x})$ [11, 31]. In one approach termed Pairwise Expectation Maximization (PWEM) [31], all instances belonging to a pair of classes are clustered and using a cluster based probability model, the confidences $P(l|\mathbf{x})$ are estimated through the EM algorithm.

Similar ideas are used in the SVM formulation by assigning different misclassification cost to different points based on the confidence in its label. Points for which the labels are more accurate, are assigned higher misclassification cost. This will lead the SVM learner not to misclassify correctly labeled points. A probabilistic scheme is proposed in [32] to find the label confidences. This scheme has been applied for text classification [32]. Given the noisy labeled training data, probabilities $P(y|\mathbf{x})$ are to be estimated for each value of \mathbf{x} and y . These probabilities are learnt by estimating parameters of a Bayes network using EM algorithm. A weighted naive Bayes classifier is finally used for classification.

6.6. Noise Tolerant Learning Algorithms

In this section we discuss some approaches whose learning strategy makes them robust to label noise. These type of algorithms are not designed to treat the noisy data differently and that is an aspect on which they can be distinguished from the algorithms discussed in the previous section. We discuss a couple of such methods here. In a later section we separately discuss some recent noise tolerant risk minimization approaches.

6.6.1. Statistical query method

Statistical query (SQ) approach [33] is among the earliest noise tolerant approaches which can be proved to learn the target concept (under some conditions) in the framework of probably Approximately Correct (PAC)

learning. It provides a framework to identify the target concept (or classifier) by estimating statistical properties of the distribution of the examples rather than directly use the examples in learning the classifier. The learning algorithm is assumed to have access to an ‘oracle’, called *STAT* which provides accurate estimates of expectations of a class of binary random variables which are indicator functions over $\mathcal{X} \times \mathcal{Y}$. (Recall that \mathcal{X} is the feature space and \mathcal{Y} is the set of class labels and that the training examples are drawn *iid* from $\mathcal{X} \times \mathcal{Y}$). The oracle provides these estimates by making use of a set of *iid* examples (and approximating expectations by sample means).

The expectations of binary random variables that the learner needs are viewed as the questions that the learner is asking and these are the statistical queries. For example: (a) what fractions of data is class-1 (b) what fraction of data from class-1 has $x_1 \geq 0.2$? It is easy to see that the answers needed here are expectations of appropriately defined indicator random variables. The oracle *STAT* is required to provide, with probability greater than $(1 - \delta)$, estimates of the expectations with an accuracy of α when given sufficient number of examples where the required number of examples can grow at most polynomially with $1/\alpha$ and $\log(1/\delta)$.

In situations where we have label noise, the oracle *STAT* does not have access to noise-free examples but only examples corrupted with label noise. The design of the algorithm involves designing queries such that using noisy examples we can still compute the required expectation, with respect to the noise-free distribution, to the required level of accuracy.

The usefulness of the SQ approach is determined by three factors: (a) an efficient SQ algorithm for learning under no noise; (b) a transformation that relates the noisy estimates to the noiseless estimates; and (c) a computationally efficient way to invert this transformation to extract the noiseless estimates.

As mentioned earlier, this is one of the methods that can be shown to learn correctly, under label noise, in a PAC sense. For example, there is a statistical queries algorithm for learning linear classifiers for certain family of radially symmetric class conditional densities and under uniform label noise [33].

The queries to be used are very much problem specific and one needs to design clever queries for each problem. Thus this is not a generic method. Also, the method can essentially take care of only uniform label noise.

6.6.2. Probabilistic approaches for noise modeling

Classification problem can also be solved via generative models. The main advantage of using generative approach is that it provides a natural way to incorporate noise process into the model. In this section, we discuss a few methods that use probabilistic models for label noise for robust learning of classifiers.

6.6.2.1. Factoring the joint probability distribution of \mathbf{x}, y, \hat{y}

Here we consider a 2-class problem with the two class labels being 0 and 1 where we have class-conditional label noise. A useful generative model can be developed by factoring the joint probability distribution $p(\mathbf{x}, y, \hat{y})$ of feature vector \mathbf{x} , ‘true’ label y and the noisy label \hat{y} as $p(\mathbf{x}, y, \hat{y}) = P(y|\hat{y})p(\mathbf{x}|y)P(\hat{y})$ [34]. In Section 6.2.1 we used the conditional distribution $P(\hat{y}|y, \mathbf{x})$ to characterize general label noise. In the class-conditional noise model, since noise rate is same for all feature vectors of one class, this conditional distribution becomes $P(\hat{y}|y)$. In the case of class-conditional noise, the noise can be specified also through the conditional distribution $P(y|\hat{y})$. This conditional distribution, $P(y|\hat{y})$, is completely specified through two parameters, γ_0, γ_1 , defined as

$$\gamma_0 = P[y = 1|\hat{y} = 0] \quad \text{and} \quad \gamma_1 = P[y = 0|\hat{y} = 1]$$

To estimate the underlying joint distribution, it is required to estimate conditional distribution $P(y|\hat{y})$ and also the class conditional densities $p(\mathbf{x}|y)$. The class conditional densities are assumed to be multivariate Gaussian with equal covariance matrices. The densities are estimated by maximizing the likelihood function. Assuming the true labels y as latent variables, maximizing the likelihood is done using EM algorithm. The following form of log likelihood is optimized using EM algorithm.

$$L(\theta) = \sum_{n=1}^N P(y|\mathbf{x}, \hat{y}) \ln p(\mathbf{x}, y|\hat{y}, \theta) \quad (6.3)$$

where the parameters, θ , consist of the means and variances of both the class conditional densities and the parameters of the conditional distribution, γ_0 & γ_1 .

The method can be kernelized and thus it actually finds kernel Fisher discriminant. However, the assumption of Gaussian class conditional densities with equal covariance matrices restricts the applicability of the method to some extent. The method is generalized to the case where each class conditional density can be a mixture of Gaussians in [35, 36].

6.6.2.2. Modeling the posterior probability $p(\hat{y}|\mathbf{x})$

Logistic regression is an approach which models the posterior probability $p(y|\mathbf{x})$ using sigmoid function: $p(y|\mathbf{x}) = 1/(1 + \exp(-yf(\mathbf{x})))$ and we take $f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$. Kernel logistic regression can learn nonlinear classifiers through kernel trick by taking $f(\mathbf{x}) = \sum_{n=1}^N \alpha_n \kappa(., \mathbf{x}_n)$, where $\kappa(., .)$ is the kernel function, α_n are some weights and \mathbf{x}_n are the training samples. Normal logistic regression assumes that the labels are not noisy. An extension of kernel logistic regression is proposed to handle the label noise in [37] where the posterior probability is modelled as

$$\begin{aligned} p(\hat{y} = k | \kappa(., \mathbf{x}), \Theta) &= \sum_{j=0}^1 p(\hat{y} = k | y = j) p(y = j | \kappa(., \mathbf{x}), \mathbf{w}) \\ &= \sum_{j=0}^1 \omega_{jk} p(y = j | \kappa(., \mathbf{x}), \mathbf{w}) \end{aligned} \quad (6.4)$$

where $\omega_{jk} = p(\hat{y} = k | y = j)$. The ω_{jk} , $j, k \in \{0, 1\}$ form the so called label flipping probability matrix Ω . Let $\Theta = \{\Omega, \mathbf{w}\}$ be the parameters to be learnt where \mathbf{w} represents all the parameters needed to represent the posterior probability when noise-free examples are available. This robust kernel logistic regression model is fit by maximizing some regularized log likelihood through the EM algorithm. A similar method is described in [38] for the application of blog comment spam detection.

6.6.3. Estimating true probabilities of error using noisy examples

Consider a 2-class problem with class-conditional label noise. If we can accurately estimate the Type-I and Type-II errors of any classifier then we can, in principle, design a Bayes classifier or a min-max classifier etc. However, we need these errors with respect to the noise-free distribution but we have examples contaminated by class-conditional noise. Recently, a method of estimating, from noisy training data, true Type 1 and Type 2 error rates of any specific classifier is proposed [6]. We briefly explain this idea below.

As defined in 6.6.2.1, let

$$\gamma_0 = P[y = 1 | \hat{y} = 0] \quad \text{and} \quad \gamma_1 = P[y = 0 | \hat{y} = 1]$$

Let p_0 and p_1 be the distributions of feature vectors of the two classes under noise-free case and let \tilde{p}_0 and \tilde{p}_1 be the distributions under the noisy case.

Under class-conditional label noise, the observed distribution, \tilde{p}_0 , of feature vectors labelled as class-0 would be a mixture of the true class conditional distributions, p_0 and p_1 and similarly for \tilde{p}_1 .

$$\tilde{p}_0 = (1 - \gamma_0)p_0 + \gamma_0 p_1$$

$$\tilde{p}_1 = (1 - \gamma_1)p_1 + \gamma_1 p_0$$

When we have $p_0 \neq p_1$ and $\gamma_0 + \gamma_1 < 1$, it turns out that we can equivalently represent the above distributions as:

$$\tilde{p}_0 = (1 - \gamma'_0)p_0 + \gamma'_0 \tilde{p}_1$$

$$\tilde{p}_1 = (1 - \gamma'_1)p_1 + \gamma'_1 \tilde{p}_0$$

where $\gamma'_0 = \frac{\gamma_0}{1 - \gamma_1}$ and $\gamma'_1 = \frac{\gamma_1}{1 - \gamma_0}$.

Since we have data from distributions, \tilde{p}_0, \tilde{p}_1 , we can, e.g., estimate the Type 1 and Type 2 error rates under these distributions. So, if we can estimate γ'_0, γ'_1 , we can estimate the error rates under p_0, p_1 distributions using only the noisy samples which are drawn from \tilde{p}_0, \tilde{p}_1 .

Then the problem reduces to the following. We have three distributions, F, G, H , related as

$$F = (1 - \nu)G + \nu H, \quad 0 \leq \nu \leq 1.$$

We have samples from F and H but do not know G . Can we estimate ν ? (For example, we can take $F = \tilde{p}_0$, $H = \tilde{p}_1$, $G = p_0$ and $\nu = \gamma'_0$). As stated, the problem is very general and ν is not an identifiable parameter. However, a good estimate for ν can be derived if we assume that G is *irreducible* with respect to H . Given any two distributions G and H , we say G is irreducible with respect to H if there exists no decomposition of the form $G = \rho H + (1 - \rho)F'$ where F' is some probability distribution and $0 < \rho \leq 1$. If G is irreducible with respect to H and H is irreducible with respect to G then they are said to be mutually irreducible. For example, if the support of one distribution is not a subset of the support of the other then the two distributions are mutually irreducible. It is shown in [6] that if we assume p_0 and p_1 to be mutually irreducible and $\gamma_0 + \gamma_1 < 1$, then it is possible to derive a good estimate for the parameters γ'_0 and γ'_1 . These would allow us to estimate true error probabilities of any classifiers based on the error rates on the noisy data.

This method is an interesting option to learn under class-conditional noise. The assumption of mutual irreducibility can be one limiting aspect of the method. Since this method is good for calculating true error rates given only noisy data, it may be particularly useful for obtaining the true error rate of a learnt classifier on the validation set even when the validation set is corrupted by class-conditional noise.

6.7. Noise Tolerant Risk Minimization

In this section we discuss some recent developments in noise-tolerant learning of classifiers through risk minimization. Many standard machine learning algorithms such as logistic regression, SVM, backpropagation for learning neural networks etc, can be viewed as risk minimization algorithms. The Bayes classifier is also a minimizer of risk. Hence, it is very relevant to ask when would the generic strategy of risk minimization be tolerant to label noise. Some general results on making risk minimization noise-tolerant are presented in [8, 39]. We discuss these results in some detail in this section.

We first briefly explain the notion of risk minimization [40, 41]. Recall that, under our notation, \mathcal{X} is the feature space and \mathcal{Y} is the set of class labels. Here, we consider only the 2-class problem. We take $\mathcal{Y} = \{+1, -1\}$ and assume that $\mathcal{X} \subset \mathbb{R}^d$. We are given a training set of examples drawn *iid* from $\mathcal{X} \times \mathcal{Y}$. Using this, we search over a family of classifiers, \mathcal{F} to find the ‘best’ classifier. We take each $f \in \mathcal{F}$ to be a real-valued function on \mathcal{X} . The class assigned by a classifier f for a feature vector \mathbf{x} would be $\text{sign}(f(\mathbf{x}))$. For example, if we are considering linear classifiers then $\mathcal{F} = \{W^T \mathbf{x} + w_0 : W \in \mathbb{R}^d, w_0 \in \mathbb{R}\}$. Taking $f(\mathbf{x})$ to be in \mathbb{R} rather than in \mathcal{Y} allows us to treat many different classification learning algorithms through a common framework.

We use the so called loss function, $L : \mathbb{R} \times \mathcal{Y} \rightarrow \mathbb{R}^+$, to assess the goodness of different classifiers. Given any random example $(\mathbf{x}, y) \in \mathcal{X} \times \mathcal{Y}$, $L(f(\mathbf{x}), y)$ tells us how well does the classifier f perform on this point. As an example, consider the 0-1 loss function defined by

$$\begin{aligned} L_{0-1}(f(\mathbf{x}), y) &= 1 \text{ if } y f(\mathbf{x}) \leq 0 \\ &= 0 \text{ otherwise} \end{aligned}$$

Here the loss is 1 whenever we make a wrong classification decision. Since we have taken $\mathcal{Y} = \{+1, -1\}$, if $f(\mathbf{x})$ has same sign as y then on (\mathbf{x}, y) , classification of f is correct; otherwise it is wrong.

Given a loss function L and a classifier f , we define the L-risk of f by

$$R_L(f) = \mathbb{E}[L(f(\mathbf{x}), y)] \quad (6.5)$$

where \mathbb{E} denotes expectation with respect to the distribution with which examples are drawn. Under the risk minimization framework, our goal is to find a classifier that has least value of risk. It is easy to see that for the 0-1 loss function, the risk is probability of misclassification. Hence the

standard Bayes classifier minimizes the risk with respect to 0-1 loss function. Many standard machine learning algorithms such as linear least-squares method, the backpropagation algorithm for learning feed forward networks, the SVM algorithm etc., can all be seen to be doing risk minimization with appropriate loss functions. Risk minimization is a generic method for supervised learning of classifiers.

As is easy to see, the risk depends on the loss function and that is why we used the term L-risk. Since we do not know the underlying probability distributions, we may not be able to compute the expectation in Eq. (6.5). Hence, we approximate the expectation by the sample average on the training data and minimize the so called empirical risk defined by

$$\hat{R}_L(f) = \frac{1}{N} \sum_{i=1}^N L(f(\mathbf{x}_i), y_i) \quad (6.6)$$

where $\{(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_n, y_N)\}$ are the *iid* training examples. Most learning algorithms essentially involve minimization of empirical risk (along with a regularization term).

Minimizing risk with respect to 0-1 loss is good because we get a classifier with minimum probability of misclassification. However, the 0-1 loss function, viewed as a function of the single variable $yf(\mathbf{x})$, is non-convex and non-smooth. Hence, the resulting optimization problem is hard. Many other loss functions such as the squared error loss (used in learning feedforward neural networks), the hinge loss (used in SVM) etc., can be thought of as convex approximations to the 0-1 loss function [42]. These loss functions are plotted (as functions of the single variable $f(\mathbf{x})y$) in Fig. 6.1.

We now formalize the notion of noise tolerant risk minimization. As in Sec. 6.2.1, let $S = \{(\mathbf{x}_i, y_{\mathbf{x}_i}), i = 1, \dots, N\}$ be the (unobservable) ideal noise-free data where each of the (\mathbf{x}_i, y_i) is drawn according to a distribution \mathcal{D} on $\mathcal{X} \times \mathcal{Y}$. The training data that is actually given to the learning algorithm is noisy and it is denoted by $S_\eta = \{(\mathbf{x}_i, \hat{y}_{\mathbf{x}_i}), i = 1, \dots, N\}$ where $\hat{y}_{\mathbf{x}_i} = -y_{\mathbf{x}_i}$ with probability $\eta_{\mathbf{x}_i}$ and is $y_{\mathbf{x}_i}$ with probability $(1 - \eta_{\mathbf{x}_i})$. (Note that we are considering a 2-class case here). Note that the probability that the label of an example in the training data is wrong is, in general, a function of the feature vector. As mentioned earlier we use $y_{\mathbf{x}}$ to denote the noise-free class label of \mathbf{x} and $\hat{y}_{\mathbf{x}}$ to denote its noisy label. We use \mathcal{D}_η to denote the joint distribution of $(\mathbf{x}, \hat{y}_{\mathbf{x}})$.

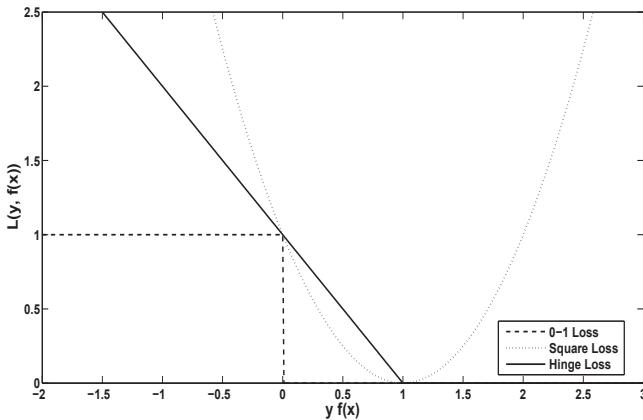


Fig. 6.1. Some convex loss functions which are used as approximations to 0-1 loss.

Now for any classifier, f , the L-risk under the noise-free case is

$$R_L(f) = \mathbb{E}_{\mathcal{D}}[L(f(\mathbf{x}), y)]$$

where the subscript (on \mathbb{E}) denotes the distribution with respect to which the expectation is taken. Let f^* be the global minimizer of $R_L(f)$. Under the noisy distribution, the L-risk of any f is

$$R_L^\eta(f) = \mathbb{E}_{\mathcal{D}_\eta}[L(f(\mathbf{x}), y)].$$

Let f_η^* be the global minimizer of $R_L^\eta(f)$. Since we are given only samples from the noisy distribution, the best we can hope for is to learn f_η^* . Note that both f^* and f_η^* depend on L , the loss function, though our notation does not explicitly show this.

We say that risk minimization under a loss function is noise-tolerant if

$$P_{\mathcal{D}}[\text{sign}(f^*(\mathbf{x})) = y_{\mathbf{x}}] = P_{\mathcal{D}}[\text{sign}(f_\eta^*(\mathbf{x})) = y_{\mathbf{x}}].$$

When this is satisfied we also say that the loss function, L , is noise-tolerant. Risk minimization under loss, L , is noise tolerant if f_η^* and f^* both have the same probability of correct classification on noise-free test data. If we are given the ideal noise-free data, then we can (in principle) learn f^* . Hence, if L is noise-tolerant then by minimizing the L-risk using noisy samples, we can learn a classifier which is as good as the one we could have learnt if we are given ideal noise-free data. Thus our definition of noise-tolerance correctly captures the level of robustness needed.

In [8] it is shown that 0-1 loss is noise tolerant for uniform noise. It is also shown to be tolerant to the most general nonuniform noise if the two classes are separable under the noise free distribution. Since Bayes classifier minimizes risk under 0-1 loss, this results shows that Bayes classifier has very good noise-tolerance properties. Recently Nettleton *et al.* [1] presented an extensive empirical study of the noise tolerance of many machine learning algorithms and found that the Naive Bayes classifier performed the best among all the algorithms they tested. The above theoretical result provides a justification for this. Learning of linear classifiers through risk minimization under squared error loss is also shown to be tolerant to uniform noise in [8]. Thus the linear least squares approach is robust to uniform noise. However, through a counter example, it is shown that squared error loss is not noise tolerant under nonuniform noise [8]. It is also shown in [8], through counter examples, that many of the convex loss functions (used in SVM, logistic regression) are not tolerant even to uniform noise. These results are generalized in [39] to derive some sufficient conditions on the loss function for it to be noise-tolerant. These are explained below.

Let us call a loss function, L , symmetric if it satisfies

$$L(f(\mathbf{x}), 1) + L(f(\mathbf{x}), -1) = K, \quad \forall \mathbf{x}, \forall f \quad (6.7)$$

for some positive constant K . It is shown in [39] that risk minimization under any symmetric loss function is tolerant to uniform noise. The 0-1 loss function is symmetric. None of the standard convex loss functions is symmetric. (This can be seen from Fig. 6.1 for the example loss functions there). However, we can have continuous (but non-convex) approximations of 0-1 loss that satisfy the above symmetry condition. A couple of examples of such loss functions are illustrated in Fig. 6.2. Here the sigmoid loss is a standard sigmoidal approximation to the 0-1 loss function and ramp loss is a kind of saturated version of the hinge loss. Both these are non-convex. And all such symmetric loss functions are tolerant to uniform noise [39]. Interestingly, it is possible to have a convex loss that satisfies the sufficient condition for noise tolerance and the corresponding risk minimization essentially amounts to a highly regularized SVM [49].

It is also shown in [39] that both the loss functions shown in Fig. 6.2 can be noise-tolerant under nonuniform noise also if the two classes are separable under the noise-free distribution and if we fix the parameter of these loss functions at a sufficiently high value. (This essentially means we need to scale the functions so that the slope of the graph in Fig. 6.2 at origin is sufficiently high). Thus these symmetric loss functions are very

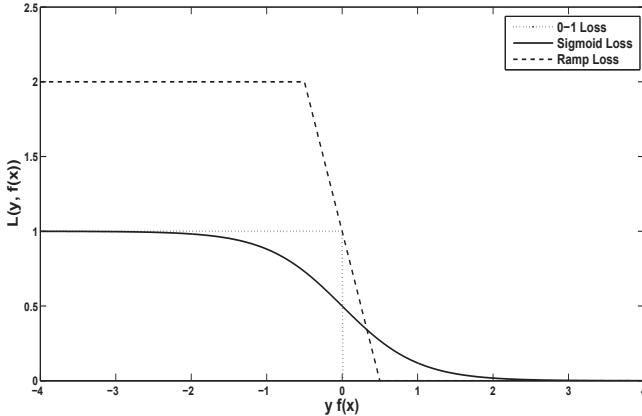


Fig. 6.2. Some symmetric loss functions which are used as approximations to 0-1 loss.

attractive from the point of view of making risk minimization noise-tolerant. Recently, it is shown that, if the class conditional densities in the noise free case belong to the generalized linear model family, the Isotron [50] can efficiently and provably learn from the corrupted sample [51].

All the above results deal with noise-tolerance under uniform or non-uniform noise. For tolerance of uniform noise, symmetry of the loss function is all that we need. For noise tolerance under nonuniform noise, we need the classes to be separable under the ideal noise-free distributions. This may not be very restrictive in many situations. However, one would like to know whether this can be relaxed in some special cases.

The class conditional noise (as explained in Sec. 6.2.1) is a special case of nonuniform noise and is a good model for label noise in many applications. Under class conditional noise, we can devise noise-tolerant risk minimization strategies where we do not need the two classes to be separable.

Suppose L is a symmetric loss function satisfying Eq. (6.7). Define a new loss function ℓ by

$$\ell(f(\mathbf{x}), 1) = L(f(\mathbf{x}), 1) \quad \text{and} \quad \ell(f(\mathbf{x}), -1) = \left(\frac{1 - \eta_1 + \eta_2}{1 - \eta_2 + \eta_1} \right) L(f(\mathbf{x}), -1)$$

where η_1, η_2 are the two noise rates under class-conditional noise. Then the minimizer of risk with loss ℓ under the noisy distribution would be the same as the minimizer of risk with loss L under the noise-free distribution. This result is proved for the case when L is 0-1 loss in [43] and for the general case of any symmetric loss function in [39]. Hence, all we need to

do is to minimize risk with respect to ℓ with the noisy examples to achieve noise tolerance under class conditional noise. We do not need the classes to be separable even under the noise-free distribution. However, to construct the loss function ℓ , we need to know the noise rates. We can, for example, estimate them using a method such as that in [6].

As mentioned earlier, minimizing risk with respect to 0-1 loss is difficult because it results in a non-convex and non-smooth objective function to minimize. An algorithm for this risk minimization, for the problem of learning linear classifiers, is proposed in [44]. This algorithm is based on the so called continuous action-set learning automata [45]. The basic idea is as follows. Suppose $W \in \mathbb{R}^d$ is the weight vector representing the linear classifier. Then the algorithm maintains a multidimensional normal density with diagonal covariance matrix in \mathbb{R}^d . The mean vector of this distribution represents our current guess on the optimal value of W . At each iteration we get a random W vector from this distribution and classify the next example pattern with this classifier and also with the classifier represented by the current mean vector of the distribution. The correctness or otherwise of these two classifications are used to update the means and variances of the normal distribution through a stochastic gradient descent algorithm. It is proved that the algorithm converges to a linear classifier that minimizes risk under 0-1 loss even under non-uniform label noise. Through simulations, the algorithm is seen to be quite effective in noise-tolerant learning of linear classifiers [8, 44]. However, it is difficult to extend this method for learning general nonlinear classifiers due to problems of computational complexity and slow convergence of the algorithm.

The sigmoidal loss is a smooth approximation to 0-1 loss and hence risk minimization would need optimization of a non-convex but smooth objective function. This can be done, for example, through a usual gradient descent type algorithm. Even though this can become computationally intensive, it is seen through simulations that risk minimization with sigmoid loss also results in noise-tolerant learning of classifiers [39].

We next consider the ramp loss. Risk minimization under ramp loss has been seen empirically to be more robust to label noise than SVM [46, 47]. The theoretical results described above tell us why it is so. The ramp loss has an interesting structure and it can be written as difference of two convex functions. This allows the use the so called DC programming methods to derive efficient algorithms for risk minimization under ramp loss [39]. Such an algorithm allows learning of nonlinear classifiers also by employing the kernel trick and it is seen to be very effective for noise-

tolerant learning of general nonlinear classifiers also [39].

The results described in this section regarding the noise tolerance properties of symmetric loss functions suggests interesting possibilities for robust learning of classifiers under label noise. Risk minimization has been a generic method that is used extensively in designing machine learning algorithms. As mentioned earlier, we would actually minimize the empirical risk (mostly along with a regularization term). However, most approaches try to use a convex approximation of the 0-1 loss so that the resulting optimization problem can be solved efficiently. The popular method of SVMs is an example of such an approach. The results in [39] indicate that it is better to use a symmetric loss function (*e.g.*, 0-1 loss, sigmoid loss or ramp loss) especially when we have to deal with label noise.

6.8. Conclusions

In this chapter, we have discussed the problem of learning good classifiers when the training examples have label noise. Learning with mislabelled training examples is hard and most standard techniques for supervised learning of classifiers perform badly under label noise. We have explained the sources of label noise to motivate why we should look for special methods so that the learnt classifiers would be good. We have explained the salient features of a representative mix of techniques for this problem. We have looked at techniques that rely on data preprocessing as well as techniques where the learning algorithm is designed to be resilient to label noise. We have also explained in detail some of the recent results in making risk minimization tolerant to label noise. This constitutes a generic method that is applicable in a variety of applications for robust learning of classifiers.

As we mentioned earlier, mislabeled data in classification problems is a common phenomena. However, there are also interesting learning problems which can be tackled by posing them as problems of learning under label noise. Suppose the only labelled examples available are for the positive class. All these are noise-free examples. But we have no labelled negative examples. However, we have access to a large set of unlabelled examples. This problem is often called learning from positive and unlabelled data. We can think of this as learning under label noise. Suppose we assign negative class to all the unlabeled points in the training data. This way all the negative class points in the unlabeled data are assigned correct labels. However, positive class points in the unlabeled data are assigned wrong labels. Thus, if we look at the whole training data, the label noise affects only the posi-

tive class. This can be thought of as a particular case of class conditional label noise. In [48], the authors show that a risk minimization technique can be successful on positive and unlabeled data only if the loss function satisfies the symmetry condition described in Eq. (6.7). Again, ramp loss is one such loss function. In their paper, they show experimentally that in the case of positive and unlabeled data, ramp loss based risk minimization performs better than SVM (hinge loss based approach).

Learning classifiers from noisy examples is an important problem of current research in Machine learning. In the years to come we would see many more developments in this area.

References

- [1] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, A study of the effect of different types of noise on the precision of supervised learning techniques, *Artificial Intelligence Review*. **33**, 275–306 (2010).
- [2] B. Frénay and M. Verleysen, Classification in the presence of label noise: a survey, *IEEE Transactions on Neural Networks and Learning Systems*. **25**, 845–869 (2014).
- [3] C. E. Brodley and M. A. Friedl, Identifying mislabeled training data, *Journal of Artificial Intelligence Research*. **11**, 131–167 (August, 1999).
- [4] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, second edn. John Wiley & Sons (2000).
- [5] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*, first edn. Springer (2006).
- [6] C. Scott, G. Blanchard, and G. Handy. Classification with asymmetric label noise: Consistency and maximal denoising. In *Conference On Learning Theory*, vol. 30, *W&CP*, pp. 489–511, JMLR (June, 2013).
- [7] R. A. S. Philip M. Long. Random classification noise defeats all convex potential boosters. In *Proceedings of the Twenty-Fifth International Conference on Machine Learning*, Helsinki, Finland (July, 2008).
- [8] N. Manwani and P. S. Sastry, Noise tolerance under risk minimization, *IEEE Transactions on Cybernetics*. **43** (3), 1146–1151 (June, 2013).
- [9] S. E. Decatur. Pac learning with constant-partition classification noise and applications to decision tree induction. In *Proceedings of the Fourteenth International Conference on Machine Learning (ICML)*, pp. 83–91, Nashville, Tennessee, USA (1997).
- [10] T. Bylander. Learning noisy linear threshold functions. Technical report, Division of Computer Science, The University of Texas at San Antonio, San Antonio, Texas - 78249 (April, 1998).
- [11] C. Bouveyron and S. Girard, Robust supervised classification with mixture models: Learning from data with uncertain labels, *Pattern Recognition*. **42** (11), 2649–2658 (2009).
- [12] T. Bylander. Learning noisy linear threshold functions in the presence of clas-

- sification noise. In *Proceedings of the seventh annual conference on Computational learning theory (COLT)*, pp. 340–347, New Brunswick, New Jersey, United States (July, 1994).
- [13] A. Blum, A. Frieze, R. Kannan, and S. Vempala. A polynomial time algorithm for learning noisy linear threshold functions. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 330–338, Burlington, Vermont (October, 1996).
 - [14] G. Stempfel, L. Ralaivola, and F. Denis. Learning from noisy data using hyperplane sampling and sample averages. Technical Report 3564, HAL-CNRS, France (2007).
 - [15] S. Fine, R. Gilad-bachrach, S. Mendelson, and N. Tishby. Noise tolerant learnability via the dual learning problem. In *NSCT* (June, 1999). (available from <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.33.2481>).
 - [16] A. Angelova, Y. Abu-Mostafa, and P. Perona. Pruning training sets for learning of object categories. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, San Diego, CA, USA (June, 2005).
 - [17] G. H. John. Robust decision trees: Removing outliers from databases. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pp. 174–179, Menlo Park, CA (1995).
 - [18] I. Guyon, N. Matin, and V. Vapnik. Discovering informative patterns and data cleaning. In *AAAI Workshop on Knowledge Discovery in Databases* (1994).
 - [19] L. Daza and E. Acuna. An algorithm for detecting noise on supervised classification. In *Proceedings of the World Congress on Engineering and Computer Science (WCECS)*, San Francisco, USA (October, 2007).
 - [20] S. Har-Peled, D. Roth, and D. Zimak. Maximum margin coresets for active and noise tolerant learning. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 836–841 (2007).
 - [21] I. W. Tsang, J. T. Kwok, and P.-M. Cheung. Core vector machines: Fast svm training on very large data sets, *Journal of Machine Learning Research*. **6**, 363–392 (2005).
 - [22] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines (and Other Kernel-Based Learning Methods)*. Cambridge University Press (2000).
 - [23] L. Xu, K. Crammer, and D. Schuurmans. Robust support vector machine training via convex outlier ablation. In *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06)*, Boston, Massachusetts (July, 2006).
 - [24] A. Karmaker and S. Kwek. A boosting approach to remove class label noise. In *Proceedings of the Fifth International Conference on Hybrid Intelligent Systems (HIS)*, pp. 206–211, Rio de Janeiro, Brazil (November, 2005).
 - [25] X. Zhu, P. Zhang, X. Wu, D. He, C. Zhang, and Y. Shi. Cleansing noisy data streams. In *Proceedings of the IEEE International Conference on Data Mining (ICDM)*, pp. 1139–1144, Pisa, Italy (2008).
 - [26] R. Jin, Y. Liu, L. Si, J. Carbonell, and A. G. Hauptmann. A new boost-

- ing algorithm using input-dependent regularizer. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, Washington, DC (August, 2003).
- [27] M. Minsky and S. Papert, *Perceptrons*, expanded edn. MIT Press, Cambridge, MA (1988).
 - [28] V. P. Roychowdhury, K.-Y. Siu, and T. Kailath, Classification of linearly nonseparable patterns by linear threshold elements, *IEEE Transactions on Neural Network*. **6** (2), 318–331 (March, 1995).
 - [29] R. Khadon and G. Wachman, Noise tolerant variants of the perceptron algorithm, *Journal of Machine Learning Research*. **8**, 227–248 (2007).
 - [30] Y. Li, H. Zaragoza, R. Herbrich, J. Shawe-Taylor, and J. S. Kandola. The perceptron algorithm with uneven margins. In *Proceedings of the Nineteenth International Conference on Machine Learning (ICML)*, pp. 379–386 (2002).
 - [31] U. Rebbapragada and C. E. Brodley. Class noise mitigation through instance weighting. In *Proceedings of the European Conference on Machine Learning (ECML)*, pp. 708–715 (2007).
 - [32] G. Ramakrishnan, K. P. Chitrapura, R. Krishnapuram, and P. Bhattacharyya. A model for handling approximate, noisy or incomplete labeling in text classification. In *Proceedings of the 22nd international conference on Machine learning (ICML)*, pp. 681–688 (2005).
 - [33] M. Kearns, Efficient noise-tolerant learning from statistical queries, *Journal of the ACM*. **45** (6), 983–1006 (November, 1998).
 - [34] N. D. Lawrence and B. Schölkopf. Estimating a kernel fisher discriminant in the presence of label noise. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pp. 306–313, Williamstown, MA, USA (2001).
 - [35] Y. Li, L. F. Wessels, and M. J. Reinders. Class noise tolerant classification based on a probabilistic noise model. In *Proc. of the 12th annual conference of the Advanced School for Computing and Imaging* (2006).
 - [36] Y. Li, L. F. A. Wessels, D. de Ridder, and M. J. T. Reinders. Classification in the presence of class noise using a probabilistic kernel fisher method, *Pattern Recognition*. **40** (12), 3349–3357 (2007). ISSN 0031-3203.
 - [37] T. Liu and D. Tao, Classification with noisy labels by importance reweighting, *CoRR*. **abs/1411.7718** (2014). URL <http://arxiv.org/abs/1411.7718>.
 - [38] A. Kantchelian, J. Ma, L. Huang, S. Afroz, A. D. Joseph, and J. D. Tygar. Robust detection of comment spam using entropy rate. In *Proceedings of the 5th ACM Workshop on Security and Artificial Intelligence (AISec)*, pp. 59–70, Raleigh, NC, USA (2012).
 - [39] A. Ghosh, N. Manwani, and P. S. Sastry, Making risk minimization tolerant to label noise, *Neurocomputing*, Vol. 160, pp. 93–107, 2015.
 - [40] D. Haussler, Decision theoretic generalizations of the PAC model for neural net and other learning applications, *Information and Computation*. **100**, 78–150 (1992).
 - [41] L. Devroye, L. Gyorfi, and G. Lugosi, *A Probabilistic Theory of Pattern Recognition*. Springer-Verlag, New York (1996).

- [42] P. L. Bartlett, M. I. Jordan, and J. D. McAuliffe, Convexity, classification and risk bounds, *Journal of the American Statistical Association*. **101** (473), 138–156 (2006).
- [43] N. Natarajan, I. Dhillon, P. Ravikumar, and A. Tewari. Learning with noisy labels. In *Advances in Neural Information Processing Systems 26*, pp. 1196–1204 (2013).
- [44] P. S. Sastry, G. D. Nagendra, and N. Manwani, A continuous-action learning automata team for noise tolerant learning of half spaces, *IEEE Transaction on System, Man and Cybernetics Part B: Cybernetics*. **40** (1), 19–28 (February, 2010).
- [45] M. A. L. Thathachar and P. S. Sastry, *Networks of Learning Automata: Techniques for Online Stochastic Optimization*. Kluwar, Boston, MA (2004).
- [46] Y. Wu and Y. Liu, Robust truncated hinge loss support vector machines, *Journal of the American Statistical Association*. **102** (479), 974–983 (Sep., 2007).
- [47] J. P. Brooks, Support vector machines with the ramp loss and the hard margin loss, *Operations Research*. **59** (2), 467–479 (May, 2011).
- [48] M. C. du Plessis, G. Niu, and M. Sugiyama. Analysis of learning from positive and unlabeled data. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 703–711 (2014).
- [49] B. van Rooyen, A. Menon and R. C. Williamson, Learning with symmetric label noise: The importance of being unhinged. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 10–18, (2015).
- [50] A. T. Kalai and R. Sastry, The Isotron algorithm: High-dimensional isotonic regression. In *Conference on Computational Learning Theory (COLT)*, (2009).
- [51] A. K. Menon and B. van Rooyen, Nagarajan Natarajan: *Learning from Binary Labels with Instance-Dependent Corruption*. CoRR abs/1605.00751 (2016).

Chapter 7

Sparse Representation for Time-Series Classification

Soheil Bahrampour¹, Nasser M. Nasrabadi² and Asok Ray¹

¹*Department of Electrical Engineering
Pennsylvania State University, University Park, PA, USA
{soheil, axr2}@psu.edu*

²*Army Research Laboratory, Adelphi, MD, USA
nasser.m.nasrabadi.civ@mail.mil*

This chapter studies the problem of time-series classification and presents an overview of recent developments in the area of feature extraction and information fusion. In particular, a recently proposed feature extraction algorithm, namely symbolic dynamic filtering (SDF), is reviewed. The SDF algorithm generates low-dimensional feature vectors using probabilistic finite state automata that are well-suited for discriminative tasks. The chapter also presents the recent developments in the area of sparse-representation-based algorithms for multimodal classification. This includes the joint sparse representation that enforces collaboration across all the modalities as well as the tree-structured sparsity that provides a flexible framework for fusion of modalities at multiple granularities. Furthermore, unsupervised and supervised dictionary learning algorithms are reviewed. The performance of the algorithms are evaluated on a set of field data that consist of passive infrared and seismic sensors.

7.1. Introduction

Unattended ground sensor (UGS) systems have been extensively used to monitor human activities for border security and target classification. Typical sensors used for this purpose are Seismic and Passive Infrared (PIR) sensors which are commonly used for target detection. Nevertheless discrimination of different types of targets from footstep signals is still a challenging problem due to environmental noise sources and locality of the sensors [1]. This study deals with the problem of target classification, and

more generally time-series classification, in two main directions, feature extraction and information fusion.

Several feature extraction methods have been proposed to generate discriminative patterns from time-series data. This includes kurtosis algorithm [2], Fourier and Wavelet analysis [3–5]. Recently, a symbolic dynamic filtering (SDF) algorithm has been proposed for feature extraction from time-series data and has shown promising results in several applications including robot motion classification and target classification [6–8]. In the SDF algorithm, the time series data are first converted into symbol sequences, and then probabilistic finite-state automata (PFSA) are constructed from these symbol sequences to compress the pertinent information into low-dimensional statistical patterns [9]. The advantage of the SDF algorithm is that it captures the local information of the signal and it is capable of mitigating noise. In this chapter, the SDF algorithm is briefly reviewed and is subsequently used for feature extraction from PIR and Seismic sensors.

This chapter also studies information fusion algorithms which is then utilized to integrate the information of the PIR and Seismic modalities. As it has been widely studied, information fusion often results in better situation awareness and decision making [10, 11]. For this purpose, several sparsity models are discussed which provide a framework for feature-level fusion. Feature-level fusion [12] is relatively less-studied topic compared to the decision-level fusion [13, 14], mainly due to the difficulty in fusing heterogeneous feature vectors. However, as it will be shown, structured sparsity priors can be used to overcome this difficulty and result in state-of-the art performance. In particular, the joint sparse representation [15] is presented which enforces collaboration across all the modalities. The tree-structured sparse representation [16, 17] is also presented that allows fusion of different modalities at multiple granularities. Moreover, unsupervised and supervised dictionary learning algorithms are discussed to train more compact dictionaries which are optimized for reconstructive or discriminative tasks, respectively [18, 19].

The rest of this chapter is organized as follows. Section 7.2 briefly describes the SDF-based feature extraction. Section 7.3 succinctly discusses the sparsity-based models for single-modal and multi-modal classification and Section 7.4 presents the dictionary learning algorithms. Section 7.5 provides a comparative study on the performance of the discussed algorithm for the application of target classification, which is followed by conclusion and recommendations for future research in Section 7.6.

7.2. Symbolic dynamic filtering for feature extraction from time-series data

An important step for time-series classification is feature extraction from sensor signals. This step can be performed using different signal processing tools including principal component analysis [20], Cepstrum [21], wavelet analysis [22], and SDF [6, 23]. It has recently been shown that SDF can result in improved classification performance by compressing the information within a time-series window into a low-dimensional feature space while preserving the discriminative pattern of the data in several applications including target detection & classification [9, 24] and prediction of lean blowout phenomena in confined combustion [25]. The detailed information and different versions of SDF can be found in earlier publications [23, 26], and this section briefly reviews a version of this algorithm which is used later for feature extraction.

The algorithm consists of a few steps. The signal space, which is approximated by the training samples, is first partitioned into a finite number of cells that are labeled as symbols. Then, PFSA are formed to represent different combinations of blocks of symbols on the symbol sequence. Finally, for a given time-series data, the state transition probability matrix is generated and the SDF feature is extracted. The pertinent steps of this procedure are described below.

Let $\{\mathbf{u}_1, \dots, \mathbf{u}_N\}$ be the set of N training time-series where $\mathbf{u}_j \in \mathbb{R}^{1 \times M}, j = 1, \dots, N$, consists of M consecutive data points. Let Σ be a set of finitely many symbols, also known as alphabet, and its cardinality denoted as $|\Sigma|$. In the partitioning step, the ensemble of training samples is divided into $|\Sigma|$ mutually exclusive and exhaustive cells. Each disjoint region forms a cell in the partitioning and is labeled with a symbol from the alphabet Σ . Consequently, each sample of a time-series is located in a particular cell and is coded with the corresponding symbol, which results in a string of symbols representing the (finite-length) time-series. There are at least two ways for performing the partitioning task: the maximum entropy partitioning and uniform partitioning [26]. The maximum entropy partitioning algorithm maximizes the entropy of the generated symbols and results in (approximately) equal number of data points in each region. Therefore, the information-rich cells of a data set are partitioned finer and those with sparse information are partitioned coarser. On the other hand, the uniform partitioning method results in equal-sized cells. Maximum entropy partitioning is adopted here. The choice of alphabet size $|\Sigma|$ largely

depends on the specific data set and can be set using cross-validation algorithms. A smaller alphabet size results in a more compressed feature vector which is more robust to the time-series noise at the cost of more information loss.

In the next step, a PFSA is used to capture the information of a given time-series. The PFSA states represent different combinations of blocks of symbols on the symbol sequence where the transition between a state to another state is governed by a transition probability matrix. Therefore, the “states” denote all possible symbol blocks within a window of certain length. For both algorithmic simplicity and computational efficiency, the D -Markov machine structure [6] has been adopted for construction of PFSA. It is noted that D -Markov machines form a proper subclass of hidden Markov models (HMM) and have been experimentally validated for applications in various fields of research (e.g., anomaly detection and robot motion classification [8]). A D -Markov chain is modeled as a statistically locally stationary stochastic process $S = \cdots s_{-1}s_0\cdots s_1\cdots$, where the probability of occurrence of a new symbol depends only on the last D symbols, i.e.,

$$P[s_n | \cdots s_{n-D} \cdots s_{n-1}] = P[s_n | s_{n-D} \cdots s_{n-1}].$$

Words of length D on a symbol string are treated as the states of the D -Markov machine before any state-merging is executed. The set of all possible states is denoted as $Q = \{q_1, q_2, \dots, q_{|Q|}\}$ and $|Q|$ is the number of (finitely many) states and $|Q| \leq |\Sigma|^D$. Here, a D -Markov Machine with the symbol block length of each state $D = 1$ is used, i.e., $|Q| = |\Sigma|$. In this case, the number of states are equal to the number of symbols, i.e., $|\mathcal{Q}| = |\Sigma|$, where the set of all possible states is denoted as $\mathcal{Q} = \{q_1, q_2, \dots, q_{|\mathcal{Q}|}\}$ and $|\mathcal{Q}|$ is the number of (finitely many) states. The transition probabilities are defined as:

$$\mathcal{P}(q_k | q_l) = \frac{S(q_l, q_k)}{\sum_{i=1,2,\dots,|\mathcal{Q}|} S(q_l, q_i)}, \forall q_k, q_l \in \mathcal{Q} \quad (7.1)$$

where $S(q_l, q_k)$ is the total count of events when q_k occurs adjacent to q_l in the direction of motion. Consequently, the state transition probability matrix of the PFSA is given as

$$\Pi = \begin{bmatrix} \mathcal{P}(q_1 | q_1) & \dots & \mathcal{P}(q_{|\mathcal{Q}|} | q_1) \\ \vdots & \ddots & \vdots \\ \mathcal{P}(q_1 | q_{|\mathcal{Q}|}) & \dots & \mathcal{P}(q_{|\mathcal{Q}|} | q_{|\mathcal{Q}|}) \end{bmatrix}. \quad (7.2)$$

By appropriate choice of partitioning, the stochastic matrix Π is irreducible and the Markov chain is ergodic, i.e., the probability of every state being

reachable from any other state within finitely many transitions must be strictly positive under statistically stationary conditions [27].

For a given time-series window, the SDF features is then constructed as the left eigenvector p corresponding to the unity eigenvalue of the stochastic matrix Π . It should be noted that Π is guaranteed to have unique unity eigenvalue. The extracted feature vector is indeed the stationary state probability vector.

7.3. Sparse representation for classification (SRC)

The SRC algorithm has been recently introduced [28] and has shown promising results in several classification applications such as robust face recognition [28], visual tracking [29], and transient acoustic signal classification [30]. Here the SRC algorithm is reviewed. Consider a C -class classification problem with N training samples from C different classes. Let $N_c, c \in \{1, \dots, C\}$, be the number of training samples for the c^{th} class and n be the dimension of the feature vector. Here n is equal to the the SDF alphabet size $|\Sigma|$. Also let $\mathbf{x}_{c,j} \in \mathbb{R}^n$ denotes the j^{th} training sample of the c^{th} class where $j \in \{1, \dots, N_c\}$. In the SRC algorithm, a *dictionary* $\mathbf{X} \triangleq [\mathbf{X}_1 \mathbf{X}_2 \dots \mathbf{X}_C] \in \mathbb{R}^{n \times N}$ is constructed by stacking the training samples where sub-dictionary $\mathbf{X}_c = [\mathbf{x}_{c,1}, \mathbf{x}_{c,2}, \dots, \mathbf{x}_{c,N_c}] \in \mathbb{R}^{n \times N_c}$ consists of the training samples for the c^{th} class, and $N = \sum_{c=1}^C N_c$ is the total number of train samples. Given a test sample $\mathbf{p} \in \mathbb{R}^n$, it is classified based on the minimum reconstruction error of it using the different classes. The underlying assumption of SRC is that a test sample from the c^{th} class lies (approximately) within the subspace formed by the training samples of the c^{th} class and can be represented using a linear combination of *a few* training samples in \mathbf{X}_c . In other words, the test sample \mathbf{p} from the c^{th} class can be represented as

$$\mathbf{p} = \mathbf{X}\boldsymbol{\alpha} + \mathbf{e}, \quad (7.3)$$

where $\boldsymbol{\alpha}$ is the coefficient vector whose entries have value 0's except for some of the entries associated with the c^{th} class, i.e. $\boldsymbol{\alpha} = [\mathbf{0}^T, \dots, \mathbf{0}^T, \boldsymbol{\alpha}_c^T, \mathbf{0}^T, \dots, \mathbf{0}^T]^T$, and \mathbf{e} is a small error/noise term due to the imperfectness of the test and training samples. For this reason the algorithm seek to obtain the sparse coefficient vector $\boldsymbol{\alpha}$ through the following ℓ_1 optimization problem:

$$\boldsymbol{\alpha}^* = \underset{\boldsymbol{\alpha}}{\operatorname{argmin}} \|\mathbf{p} - \mathbf{X}\boldsymbol{\alpha}\|_{\ell_2} + \lambda \|\boldsymbol{\alpha}\|_{\ell_1}, \quad (7.4)$$

with ℓ_1 -norm defined as $\|\boldsymbol{\alpha}\|_{\ell_1} = \sum_{j=1}^N |\alpha_j|$ and λ is a regularization parameter. The solution of the above optimization problem is sparse provided that the regularization parameter λ is chosen sufficiently large. After the solution of the optimization problem is found, the test sample \mathbf{p} is classified by comparing the reconstruction errors of different classes. For this purpose, let $\delta_c(\boldsymbol{\alpha}) \in \mathbb{R}^N$ be a vector whose only non-zero elements are those entries in $\boldsymbol{\alpha}$ that are associated with class c in \mathbf{X} , i.e. $\delta_c(\boldsymbol{\alpha}) = [\mathbf{0}^T, \dots, \mathbf{0}^T, \boldsymbol{\alpha}_c^T, \mathbf{0}^T, \dots, \mathbf{0}^T]^T$. The label of the test data is then predicted using

$$c^* = \operatorname{argmin}_c \|\mathbf{p} - \mathbf{X}\delta_c(\boldsymbol{\alpha}^*)\|_{\ell_2}. \quad (7.5)$$

7.3.1. Joint sparse representation classification

In many applications including target classification, there are several sources of information that should be fused to make an optimal classification decision. It is well-known that information fusion of sensors can generally result in better situation awareness and decision making [10]. Many algorithms have been proposed for sensor fusion in the classification problem. These methods can generally be categorized in two sets, feature fusion [12] and classifier fusion [14] algorithms. Classifier fusion algorithms aggregate the decisions from different classifiers which are individually built based on different sources. Different methods of decision fusion include majority vote [31], fuzzy logic [32] and statistical inference [11]. For example, in the context of the sparse representation classification for target classification, the reconstruction error generated by using PIR and Seismic features individually can be combined, after proper normalization, for a fused decision. This is also known as holistic sparse representation classification (HSRC) [33]. While classifier fusion is a well-studied topic, feature level fusion is a relatively less-studied, specifically for fusing heterogeneous source of information due to the incompatibility of feature sets [34]. Feature level fusion using sparse representation has also been recently introduced and has shown promising results [35–38].

Among different sparsity based algorithms for information fusion, joint sparse representation classification (JSRC) is probably the most cited approach [21, 30, 39]. In JSRC, multiple observations of a pattern using different modalities are simultaneously represented by a few training samples. Consider the C -class classification problem and let $\mathcal{S} \triangleq \{1, \dots, S\}$ be a finite set of available modalities. Similar to the previous section, let

$N = \sum_{c=1}^C N_c$ be the number of the training samples from each modalities, where N_c is the number of training samples in the c^{th} class. Also let $n^s, s \in \mathcal{S}$, be the dimension of the feature vector for the s^{th} modality and $\mathbf{x}_{c,j}^s \in \mathbb{R}^{n^s}$ denote the j^{th} sample of the s^{th} modality that belongs to the c^{th} class, $j \in \{1, \dots, N_c\}$. In JSRC, S dictionaries $\mathbf{X}^s \triangleq [\mathbf{X}_1^s \mathbf{X}_2^s \dots \mathbf{X}_C^s] \in \mathbb{R}^{n^s \times N}, s \in \mathcal{S}$, are constructed from the (normalized) training samples, where the class-wise sub-dictionary $\mathbf{X}_c^s \triangleq [\mathbf{x}_{c,1}^s, \mathbf{x}_{c,2}^s, \dots, \mathbf{x}_{c,N_c}^s] \in \mathbb{R}^{n^s \times N_c}$ consists of samples from the c^{th} class and s^{th} modality.

Given a multimodal test sample $\{\mathbf{p}^s\}$, where $\mathbf{p}^s \in \mathbb{R}^{n^s}, s \in \mathcal{S}$, the assumption is that the test sample \mathbf{p}^s from the c^{th} class lies approximately within the subspace formed by the training samples of the c^{th} class and can be approximated (or reconstructed) from *a few* number of training samples in \mathbf{X}_c^s [28], similar to the SRC algorithm. In other words, if the test sample \mathbf{p}^s belongs to the c^{th} class, it is represented as:

$$\mathbf{p}^s = \mathbf{X}^s \boldsymbol{\alpha}^s + \mathbf{e}, \quad (7.6)$$

where $\boldsymbol{\alpha}^s \in \mathbb{R}^N$ is a coefficient vector whose entries are mostly 0's except for some of the entries associated with the c^{th} class, i.e., $\boldsymbol{\alpha}^s = [\mathbf{0}^T, \dots, \mathbf{0}^T, \boldsymbol{\alpha}_c^T, \mathbf{0}^T, \dots, \mathbf{0}^T]^T$, and \mathbf{e} is a small error term due to imperfectionness of the samples. In addition, JSRC recognizes the relation between different modalities representing the same event and enforces collaboration among them to make a joint decision. This is achieved by constraining the coefficient vectors from different modalities to have the same sparsity pattern. Consequently, the same training samples from different modalities are used to reconstruct the test data. To illustrate the idea, consider the target classification application with PIR and Seismic sensors. Let \mathbf{p}^1 and \mathbf{p}^2 be the feature vectors extracted from PIR and Seismic sensors, respectively. Also let the multimodal test sample belongs to the c^{th} class. Using the idea of sparse representation discussed in previous Section, test samples can be reconstructed using a linear combination of atoms in \mathbf{X}^1 and \mathbf{X}^2 , i.e., $\mathbf{p}^1 = \mathbf{X}^1 \boldsymbol{\alpha}^1 + \mathbf{e}^1, \mathbf{p}^2 = \mathbf{X}^2 \boldsymbol{\alpha}^2 + \mathbf{e}^2$, where $\boldsymbol{\alpha}^1$ and $\boldsymbol{\alpha}^2$ are the coefficient vectors whose entries have value 0's except for some of the entries associated with the c^{th} class, and \mathbf{e}^1 and \mathbf{e}^2 are small error terms. Let \mathcal{I}^1 and \mathcal{I}^2 be two index sets corresponding to non-zero rows of $\boldsymbol{\alpha}^1$ and $\boldsymbol{\alpha}^2$, respectively. In JSRC algorithm, it is further assumed that \mathbf{p}^1 and \mathbf{p}^2 can be reconstructed from the same training samples corresponding to dictionaries \mathbf{X}^1 and \mathbf{X}^2 with possibly different coefficients because they belong to the same event and therefore $\mathcal{I}^1 = \mathcal{I}^2$. In other words, $A = [\boldsymbol{\alpha}^1, \boldsymbol{\alpha}^2]$ is a row-sparse matrix with only a few non-zeros rows. In general, the coeffi-

cient matrix $\mathbf{A} = [\boldsymbol{\alpha}^1, \dots, \boldsymbol{\alpha}^S] \in \mathbb{R}^{N \times S}$, where $\boldsymbol{\alpha}^s$ is the sparse coefficient vector for reconstructing \mathbf{p}^s , is recovered by solving the following ℓ_1/ℓ_2 joint optimization problem:

$$\underset{\mathbf{A}=[\boldsymbol{\alpha}^1, \dots, \boldsymbol{\alpha}^S]}{\operatorname{argmin}} f(\mathbf{A}) + \lambda \|\mathbf{A}\|_{\ell_1/\ell_2}, \quad (7.7)$$

where $f(\mathbf{A}) \triangleq \frac{1}{2} \sum_{s=1}^S \|\mathbf{p}^s - \mathbf{X}^s \boldsymbol{\alpha}^s\|_{\ell_2}^2$ is the reconstruction error, $\lambda > 0$ is a regularization parameter, and ℓ_1/ℓ_2 norm is defined as $\|\mathbf{A}\|_{\ell_1/\ell_2} = \sum_{j=1}^N \|\mathbf{a}_j\|_{\ell_2}$ in which \mathbf{a}_j 's are row vectors of \mathbf{A} . The above optimization problem encourages sharing of patterns across related observations which results in the solution \mathbf{A} to have a common support at the column level [21]. The solution can be obtained by using the efficient alternating direction method of multipliers [40].

Again let $\delta_c(\alpha) \in \mathbb{R}^N$ be a vector indication function in which the rows corresponding to c^{th} class are retained and the rest are set to zeros. Similar to the SRC algorithm, the test data is classified using the class-specific reconstruction errors as:

$$c^* = \underset{c}{\operatorname{argmin}} \sum_{s=1}^S \|\mathbf{p}^s - \mathbf{X}^s \delta_c(\boldsymbol{\alpha}^{s*})\|_{\ell_2}^2, \quad (7.8)$$

where $\boldsymbol{\alpha}^{s*}$'s are optimal solutions of (7.7).

7.3.2. Tree-structured sparse representation classification

The joint sparsity assumption of JSRC may be too stringent for applications in which not all the different modalities are equally important for classification. Recently tree-structured sparsity [16, 17] is proposed to provide a flexible framework for information fusion. It uses the prior knowledge in grouping different modalities by encoding them in a tree and allows different modalities to be fused at multiple granularity. The leaf nodes represent individual modalities in the tree and the internal nodes represent different grouping of the modalities. A tree-structured groups of modalities $\mathcal{G} \subseteq (2^S \setminus \emptyset)$ is defined as a collection of subsets of the set of modalities S such that $\bigcup_{g \in \mathcal{G}} g = S$ and $\forall g, \tilde{g} \in \mathcal{G}, (g \cap \tilde{g} \neq \emptyset) \Rightarrow ((g \subseteq \tilde{g}) \vee (\tilde{g} \subseteq g))$. It is assumed here that \mathcal{G} is ordered according to relation \preceq which is defined as $(g \preceq \tilde{g}) \Rightarrow ((g \subseteq \tilde{g}) \vee (g \cap \tilde{g} = \emptyset))$.

Given a tree-structured collection \mathcal{G} of groups and a multimodal test sample $\{\mathbf{p}^s\}$, the tree-structured sparse representation classification

(TSRC) solves the following optimization problem:

$$\underset{\mathbf{A}=[\boldsymbol{\alpha}^1, \dots, \boldsymbol{\alpha}^S]}{\operatorname{argmin}} f(\mathbf{A}) + \lambda \Omega(\mathbf{A}), \quad (7.9)$$

where $f(\mathbf{A})$ is defined the same as in Eq. (7.7), and the tree-structured sparsity prior $\Omega(\mathbf{A})$ is defined as:

$$\Omega(\mathbf{A}) \triangleq \sum_{j=1}^N \sum_{g \in \mathcal{G}} \omega_g \|\mathbf{a}_{jg}\|_{\ell_2}. \quad (7.10)$$

In Eq. (7.10), ω_g is a positive weight for group g and \mathbf{a}_{jg} is a $(1 \times S)$ row vector whose coordinates are equal to the j^{th} row of \mathbf{A} for indices in the group g , and 0 otherwise. An accelerated algorithm can be used to efficiently solve the optimization problem [17].

The above optimization problem results in \mathbf{A}^* that has a common support at the group level and the resulting sparsity is dependent on the relative weights ω_g of different groups [16]. In the special case where \mathcal{G} consists of only one group, containing all modalities, then Eq. (7.9) reduces to that of JSRC in Eq. (7.7). On the other hand, if \mathcal{G} consists of only singleton sets of individual modalities, no common sparsity pattern is sought across the modalities and the optimization problem of Eq. (7.9) reduces to S separate ℓ_1 optimization problems. The tree-structured sparsity prior provides flexibility in the expense of the need to apiori select the weights $\omega_g, g \in \mathcal{G}$, which will be discussed in more details in the next section.

7.4. Dictionary learning

In the discussed sparsity based classification algorithms, SRC, JSRC and TSRC, the dictionary is constructed by stacking the training samples. In contrast to the conventional classification algorithms, above algorithms do not have a “training” step and most of the computation is performed at the test time, i.e., an optimization problem needs to be solved for each test sample. These results in at least two difficulties. First, the computational cost of the optimization problem becomes more and more expensive as the number of training samples increases. Second, the dictionary constructed by stacking the training samples is not optimal neither for the reconstructive tasks [41] nor the discriminative tasks [18]. Recently it has been shown that *dictionary learning* can overcome the above limitations and significantly improve the performance in several applications including image restoration [42], face recognition [43] and object recognition [44, 45].

In contrast to principal component analysis and its variants, dictionary learning algorithms generally do not impose orthogonality condition and are more flexible allowing to be well-tuned to the training data. Moreover, the size of the learned dictionaries are usually smaller than the number of training samples [46, 47] and are more appropriated for real-time applications. Dictionary learning algorithms can generally be categorized into two groups: unsupervised and supervised which are discussed in the following sections.

7.4.1. Unsupervised dictionary learning

Let $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N] \in \mathbb{R}^{n \times N}$ be the collection of N (normalized) training samples. In an unsupervised setting, the dictionary $\mathbf{D} \in \mathbb{R}^{n \times d}$ is obtained irrespective of the class labels of the training samples by minimizing of the following reconstructive cost [43]:

$$g_N(\mathbf{D}) \triangleq \frac{1}{N} \sum_{i=1}^N l_u(\mathbf{x}_i, \mathbf{D}), \quad (7.11)$$

over the regularizing convex set $\mathcal{D} \triangleq \{\mathbf{D} \in \mathbb{R}^{n \times d} \mid \|\mathbf{d}_k\|_{\ell_2} \leq 1, \forall k = 1, \dots, d\}$, where \mathbf{d}_k is the k^{th} column, or atom, of the dictionary. The loss l_u is defined as

$$l_u(\mathbf{x}, \mathbf{D}) \triangleq \min_{\boldsymbol{\alpha} \in \mathbb{R}^d} \|\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}\|_{\ell_2}^2 + \lambda \|\boldsymbol{\alpha}\|_{\ell_1}, \quad (7.12)$$

which is the optimal value of the sparse coding problem. It is usually preferred to find the dictionary by minimizing an expected risk, rather than the perfect minimization of the empirical cost for the generalization purpose [48]. A parameter-free online algorithm has also been proposed [41] to find the dictionary \mathbf{D} as the minimizer of the following stochastic cost over the convex set \mathcal{D} :

$$g(\mathbf{D}) \triangleq \mathbb{E}_{\mathbf{x}} [l_u(\mathbf{x}, \mathbf{D})], \quad (7.13)$$

where it is assumed that the data \mathbf{x} is drawn from a finite probability distribution. The trained dictionary can then be integrated into the SRC algorithm.

The trained dictionary can also be used for feature extraction. In this setting, the sparse code $\boldsymbol{\alpha}^*$, generated as a solution of (7.12), is used as a latent feature vector representing the input signal \mathbf{x} in the classical expected risk optimization for training a classifier [19]:

$$\min_{\mathbf{w} \in \mathcal{W}} \mathbb{E}_{y, \mathbf{x}} [l(y, \mathbf{w}, \boldsymbol{\alpha}^*)] + \frac{\nu}{2} \|\mathbf{w}\|_{\ell_2}^2, \quad (7.14)$$

where y is the ground truth class label associated with the input \mathbf{x} , \mathbf{w} is the classifier parameters, ν is a regularizing parameter, and l is a convex loss function that measures how well one can predict y given $\boldsymbol{\alpha}^*$ and \mathbf{w} . Note that in Eq. (7.14), the dictionary \mathbf{D} is optimized independent of the classifier and class label.

7.4.2. Supervised dictionary learning

The dictionary trained in the unsupervised setting is not optimal for classification. In a supervised formulation, the class labels can be used to train a discriminative dictionary. In the most straightforward extension from the unsupervised setting, the dictionary \mathbf{D}^* can instead be obtained by learning class-specific sub-dictionaries $\mathbf{D}_c^*, c = 1, \dots, C$, in a C -class classification problem using the formulation of Eq. (7.13) by sampling the input from the corresponding c -th class population. The overall dictionary is then constructed as $\mathbf{D}^* = [\mathbf{D}_1^* \dots \mathbf{D}_C^*]$ [43]. The C different sub-dictionaries are trained independently and some of the sub-dictionaries can possibly share similar atoms that may adversely affect the discrimination performance. An *incoherence* term has been proposed to be added to the cost function to make the derived dictionaries independent [49]. In another formulation, a discriminative term is added to the reconstruction error and the overall cost is minimized [18, 50]. More recently, it has been shown that better performance can be obtained by learning the dictionary in a task-driven formulation [19]. In the task-driven dictionary learning, the optimal dictionary and classifier parameters are obtained jointly by solving the following optimization problem [19]:

$$\min_{\mathbf{D} \in \mathcal{D}, \mathbf{w} \in \mathcal{W}} \mathbb{E}_{y, \mathbf{x}} [l_{su}(y, \mathbf{w}, \boldsymbol{\alpha}^*(\mathbf{x}, \mathbf{D}))] + \frac{\nu}{2} \|\mathbf{w}\|_{\ell_2}^2. \quad (7.15)$$

The learned task-driven dictionary has been shown to result in a superior performance compared to the unsupervised setting [19]. In this setting, the sparse codes are indeed the optimized latent features for the classifier. While the above dictionary learning algorithms are mostly developed for single-modal scenarios, it has been shown that learning multimodal dictionaries under structured sparsity priors can be beneficial [51, 52].

7.5. Results

This section presents the results of target classification on a real data set, which was generated from field data using one passive infrared (PIR) and

three seismic sensors. The time-series are classified to predict two targets: (i) human walking alone, and (ii) animal led by a walking human. These targets moved along an approximately 150 meters long trail and returned along the same trail to the starting point; all targets passed by the sensor sites at a distance of approximately 5 meters. Signals from both PIR and seismic sensors were acquired at a sampling frequency of 10 kHz and each test was conducted over a period of approximately 50 seconds. The subset of data used here consists of two days data. Day 1 includes 47 human targets and 35 animal-led-by-human targets while the corresponding numbers for Day 2 are 32 and 34, respectively. A two-way cross-validation is used to assess the performance of the classification algorithms, i.e., Day 1 data is used for training and Day 2 is used as test data and vice versa. The reported results are the average of the results on two different test data sets.

The alphabet size $|\Sigma|$ for the SDF feature extraction algorithm is chosen to be 30 as has been reported optimal for target classification in previous study [9]. The regularization parameters, and the dictionary size in cases where dictionary learning is used, are chosen based on minimizing the classifications cost on a small validation set.

For TSRC, the tree-structured set of groups is selected to be $\mathcal{G} = \{g_1, g_2, g_3, g_4, g_5, g_6\} = \{\{1\}, \{2\}, \{3\}, \{1, 2, 3\}, \{4\}, \{1, 2, 3, 4\}\}$ where 1, 2 and 3 refer to the seismic channels and 4 refers to the PIR channel. The relative weights for different groups are chosen to satisfy $\omega_{g_1} = \omega_{g_2} = \omega_{g_3} = \omega_{g_5} = 0.001\omega_{g_4}$ and $\omega_{g_6} = 10\omega_{g_4}$. This leaves the tuning to be done using only one parameter ω_{g_3} which is selected using validation set. The underlying assumption is that the three seismic channels are correlated and therefore, the weight for group g_4 is selected to be relatively big compared to the weights for individual seismic sources to encourage joint sparsity between the three seismic channels. However, the weight for g_6 is the biggest one which encourages collaboration among all sources. Although the value of correlation between different modalities are different, but the tree-structured sparsity allows collaboration at different granularities. After the relative relations of different weights are selected apriori, the value of ω_{g_3} is chosen using cross validation. It is noted that the results of the tree-structured sparsity is not sensitive to the exact values of the weights and similar results are obtained with different set of weights as long as the underlying prior information is correctly formulated.

A straightforward way of utilizing the unsupervised and supervised (single-modal) dictionary learning algorithms for multimodal classification is to train independent dictionaries and classifiers for each modality and

then combine the individual scores for a fused decision. The corresponding algorithms are referred as UDL and SDL. This way of fusion is equivalent to using the ℓ_{11} norm on \mathbf{A} , instead of ℓ_{12} norm, in Eq. (7.7) which does not enforce row sparsity in the sparse coefficients and is a decision-level fusion algorithm. The number of dictionary atoms for UDL and SDL is chosen to be 20, 10 per class, and the quadratic cost [19] is used. The dictionaries for JSRC and JDSRC are constructed using all available training samples.

The performances of the presented classification algorithms under different sparsity priors are also compared with those of the several state-of-the-art decision-level and feature-level fusion algorithms. For decision level fusion algorithms linear support vector machine (SVM) [20] and logistic regression (LR) [20] classifiers are trained on individual modalities and the fused decision is obtained by combining the score of individual modalities. These approaches are abbreviated as *SVM-Sum* and *LR-Sum*, respectively. The performance of the algorithms are also compared with feature-level fusion methods including the holistic sparse representation classifier (HSRC) [33], the joint dynamic sparse representation classifier (JDSRC) [33], relaxed collaborative representation (RCR) [38], and multiple kernel learning (MKL) [53]. The HSRC is a simple modification of SRC for multimodal classification in which the feature vectors from different sources are concatenated into a longer feature vector and SRC is applied on the concatenated feature vectors. JDSRC and RCR are also recently applied to a number of applications with better performance than JSRC. For the MKL algorithm, linear, polynomial, and RBF kernels are used.

Table 7.1 summarizes the average human detection rate (HDR), human false alarm rate (HFAR), and correct classification rates (CCR) obtained using different multimodal classification algorithms. As seen, the sparsity based feature-level fusion algorithm, JDSRC, JSRC, and TSRC have resulted in better performances than the competing algorithms with TSRC achieving the best classification result. Moreover, it is seen that the structured sparsity prior has indeed resulted in improved performance compared to the HSRC algorithm. While the SDL algorithm result in similar performance compared to the counterpart HSRC algorithm, it should be noted that SDL enjoys more compact dictionaries. Developing multimodal dictionaries using structured sparsity [51] can potentially improve the results which is a topic of future research.

Table 7.1. Results of target classification obtained by different multimodal classification algorithms by fusing 1 PIR and 3 seismic sensors data. HDR: Human Detection Rate, HFAR: Human False Alarm Rate, CCR: Correct Classification Rate.

Classification Algorithm	HDR	HFAR	CCR (%)
SVM-Sum	0.94	0.13	91.22
LR-Sum	0.97	0.13	92.57
RCR	0.94	0.12	91.22
MKL	0.94	0.08	93.24
HSRC	0.96	0.10	93.24
JSRC	1.00	0.12	94.59
JDSRC	0.97	0.08	94.59
TSRC	1.00	0.09	95.65
UDL	0.92	0.13	89.86
SDL	0.94	0.08	93.24

7.6. Conclusions

This chapter has presented an overview of symbolic dynamic filtering as a feature extraction algorithm for time-series data. The recent developments in the area of sparse representation for multimodal classification have also been presented. For this purpose, structured sparsity priors, which enforce collaboration across different modalities, are studies. In particular, the tree-structured sparsity model allows extraction of cross-correlated information among multiple modalities at different granularities. Finally, unsupervised and supervised dictionary learning algorithms were reviewed. The performances of the discussed algorithms are evaluated for the application of target classification. The results show that the feature fusion algorithms using sparsity models achieve superior performance as compared to the counter-part decision-fusion algorithms. An interesting topic of future research includes development of multimodal dictionary learning algorithms under different structured sparsity priors.

References

- [1] R. A. Gramann, M. Bennett, and T. D. Obrien, Vehicle and personnel detection using seismic sensors, *Sensors, C3I, Information, and Training Technologies for Law Enforcement*. **3577**(1), 7485 (1999).
- [2] J. Altmann, Acoustic and seismic signals of heavy military vehicles for cooperative verification, *Proc. of the IEEE*. **273**(4-5), 713740 (2004).
- [3] Y. Tian and H. Qi. Target detection and classification using seismic signal

- processing in unattended ground sensor systems. In *ICASSP*, pp. 4172–4172 (2002).
- [4] D. Li, K. D. Wong, Y. H. Hu, and A. M. Sayeed, Detection, classification, and tracking of targets, *IEEE SPM*. **19**(2), 17–29 (2002).
 - [5] G. Succi, D. Clapp, R. Gampert, and G. Prado, Footstep detection and tracking, *Unattended Ground Sensor Technologies and Applications III*. **4393**(1), 22–29 (2001).
 - [6] A. Ray, Symbolic dynamic analysis of complex systems for anomaly detection, *Signal Processing*. **84**(7), 1115–1130 (July, 2004).
 - [7] X. Jin, S. Sarkar, A. Ray, S. Gupta, and T. Damarla, Target detection and classification using seismic and PIR sensors, *IEEE SJ*. **12**(6), 1709–1718 (2012).
 - [8] G. Mallapragada, A. Ray, and X. Jin, Symbolic dynamic filtering and language measure for behavior identification of mobile robots, *IEEE TSMC*. **42**(3), 647–659 (2012).
 - [9] S. Bahrampour, A. Ray, S. Sarkar, T. Damarla, and N. Nasrabadi, Performance comparison of feature extraction algorithms for target detection and classification, *Pattern Recognition Letters*. pp. 2126–2134 (2013).
 - [10] D. L. Hall and J. Llinas, An introduction to multisensor data fusion, *Proc. IEEE*. **85**(1), 6–23 (January, 1997).
 - [11] H. Wu, M. Siegel, R. Stiefelhagen, and J. Yang, Sensor fusion using Dempster-Shafer theory. In *Proc. 19th IEEE Instrum. and Meas. Technol. Conf. (IMTC)*, pp. 7–12 (2002).
 - [12] A. Ross and R. Govindarajan, Feature level fusion using hand and face biometrics. In *SPIE proc. series*, pp. 196–204 (2005).
 - [13] S. Pirooz Azad, S. Bahrampour, B. Moshiri, and K. Salahshoor, New fusion architectures for performance enhancement of a pca-based fault diagnosis and isolation system. In *SAFEPROCESS*, pp. 852–857 (2009).
 - [14] D. Ruta and B. Gabrys, An overview of classifier fusion methods, *Comput. and Inf. syst.* **7**(1), 1–10 (2000).
 - [15] Z. Kang, K. Grauman, and F. Sha, Learning with whom to share in multi-task feature learning. In *Proc. 26th IEEE Int. Conf. Mach. Learning (ICML)*, pp. 521–528 (2011).
 - [16] S. Kim and E. Xing, Tree-guided group lasso for multi-task regression with structured sparsity, *arXiv:0909.1373* (2009).
 - [17] S. Bahrampour, A. Ray, N. M. Nasrabadi, and W. K. Jenkins, Quality-based multimodal classification using tree-structured sparsity. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, pp. 4114–4121 (2014).
 - [18] J. Mairal, F. Bach, A. Zisserman, and G. Sapiro, Supervised dictionary learning. In *Advances Neural Inform. Process. Syst. (NIPS)*, pp. 1033–1040 (2008).
 - [19] J. Mairal, F. Bach, and J. Ponce, Task-driven dictionary learning, *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(4), 791–804 (Apr., 2012).
 - [20] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer (2006).
 - [21] N. H. Nguyen, N. M. Nasrabadi, and T. D. Tran, Robust multi-sensor classification via joint sparse representation. In *Proc. 14th Int. Conf. Information Fusion (FUSION)*, pp. 1–8 (2011).

- [22] S. Mallat, *A Wavelet Tour of Signal Processing: The Sparse Way*, 3rd ed. Academic Press (2008).
- [23] K. Mukherjee and A. Ray, State splitting and merging in probabilistic finite state automata for signal representation and analysis, *Signal Processing*. **104**, 105–119 (2014).
- [24] B. Smith, P. Chattopadhyay, A. Ray, S. Phoha, and T. Damarla. Performance robustness of feature extraction for target detection and classification. In *Proc. American Control Conf.*, pp. 3814–3819 (June, 2014).
- [25] S. Sarkar, A. Ray, A. Mukhopadhyay, and S. Sen, Dynamic data-driven prediction of lean blowout in a swirl-stabilized combustor, *Intl. J Spray and Combustion Dynamics*. **7**(3), 209–242 (2015).
- [26] V. Rajagopalan and A. Ray, Symbolic time series analysis via wavelet-based partitioning, *Signal Processing*. **86**(11), 3309–3320 (2006).
- [27] A. Berman and R. Plemmons, *Nonnegative Matrices in the Mathematical Sciences*. SIAM (1994).
- [28] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, Robust face recognition via sparse representation, *IEEE Trans. Pattern Anal. Mach. Intell.* **31**(2), 210–227 (Feb., 2009).
- [29] X. Mei and H. Ling, Robust visual tracking and vehicle classification via sparse representation, *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(11), 2259–2272 (Nov., 2011).
- [30] H. Zhang, Y. Zhang, N. M. Nasrabadi, and T. S. Huang, Joint-structured-sparsity-based classification for multiple-measurement transient acoustic signals, *IEEE Trans. Syst., Man, Cybern.* **42**(6), 1586–98 (Dec., 2012).
- [31] Y. A. Zuev and S. Ivanov, The voting as a way to increase the decision reliability, *Journal of the Franklin Institute*. **336**(2), 361–378 (1999).
- [32] T. M. Chen and R. C. Luo, Multilevel multiagent based team decision fusion for autonomous tracking system, *Mach. Intell. Robot. Control.* **1**(2), 63–69 (1999).
- [33] H. Zhang, N. M. Nasrabadi, Y. Zhang, and T. S. Huang. Multi-observation visual recognition via joint dynamic sparse representation. In *Proc. IEEE Conf. Comput. Vision (ICCV)*, pp. 595–602 (2011).
- [34] A. Rattani, D. R. Kisku, M. Bicego, and M. Tistarelli. Feature level fusion of face and fingerprint biometrics. In *Proc. 1st IEEE Int. Conf. Biometrics: Theory, Applicat., and Syst.*, pp. 1–6 (2007).
- [35] S. Shekhar, V. Patel, N. M. Nasrabadi, and R. Chellappa, Joint sparse representation for robust multimodal biometrics recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* **36**(1), 113–126 (Jan., 2013).
- [36] U. Srinivas, H. Mousavi, C. Jeon, V. Monga, A. Hattel, and B. Jayarao, Simultaneous sparsity model for histopathological image representation and classification, *IEEE Trans. Med. Imag.* **33**(5), 1163 – 1179 (May, 2014).
- [37] H. S. Mousavi, V. Srinivas, U. and Monga, Y. Suo, M. Dao, and T. D. Tran. Multi-task image classification via collaborative, hierarchical spike-and-slab priors. In *IEEE Intl. Conf. Image Processing (ICIP)*, pp. 4236–4240 (2014).
- [38] M. Yang, L. Zhang, D. Zhang, and S. Wang. Relaxed collaborative representation for pattern classification. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, pp. 2224–2231 (2012).

- [39] S. Shekhar, V. M. Patel, N. M. Nasrabadi, and R. Chellappa. Joint sparsity-based robust multimodal biometrics recognition. In *European Conf. Comput. Vision*, pp. 365–374 (2012).
- [40] J. Yang and Y. Zhang, Alternating direction algorithms for ℓ_1 -problems in compressive sensing, *SISC*. **33**(1), 250–278 (2011).
- [41] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, Online dictionary learning for sparse coding, *Proc. 26th Annu. Int. Conf. Mach. Learning (ICML)*. pp. 689–696 (2009).
- [42] J. Mairal, M. Elad, and G. Sapiro, Sparse representation for color image restoration, *IEEE Trans. Image Process.* **17**(1), 53–69 (Jan., 2008).
- [43] M. Yang, L. Zhang, J. Yang, and D. Zhang. Metaface learning for sparse representation based face recognition. In *Proc. IEEE Conf. Image Process. (ICIP)*, pp. 1601–1604 (2010).
- [44] Y. L. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning mid-level features for recognition. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, pp. 2559–2566 (2010).
- [45] Z. Jiang, Z. Lin, and L. S. Davis, Label consistent K-SVD: Learning a discriminative dictionary for recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* **35**(11), 2651–2664 (Nov., 2013).
- [46] M. Aharon, M. Elad, and A. Bruckstein, K-SVD : An algorithm for designing overcomplete dictionaries for sparse representation, *IEEE Trans. Signal Process.* **54**(11), 4311–4322 (Nov., 2006).
- [47] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, Online learning for matrix factorization and sparse coding, *The J. of Mach. Learning Research.* **11**, 19–60 (2010).
- [48] L. Bottou and O. Bousquet. The trade-offs of large scale learning. In *Advances Neural Inform. Process. Syst. (NIPS)*, pp. 161–168 (2007).
- [49] I. Ramirez, P. Sprechmann, and G. Sapiro. Classification and clustering via dictionary learning with structured incoherence and shared features. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, pp. 3501–3508 (2010).
- [50] Q. Zhang and B. Li. Discriminative K-SVD for dictionary learning in face recognition. In *Proc. IEEE Conf. Comput. Vision and Pattern Recognition (CVPR)*, pp. 2691–2698 (2010).
- [51] S. Bahrampour, N. M. Nasrabadi, A. Ray, and W. K. Jenkins, Multimodal task-driven dictionary learning for image classification, *arXiv:1502.01094* (2015).
- [52] S. Bahrampour, N. M. Nasrabadi, A. Ray, and W. K. Jenkins. Kernel task-driven dictionary learning for hyperspectral image classification. In *Proc. IEEE Intl. Conf. Acoustics, Speech and Signal Process. (ICASSP)* (2015).
- [53] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet, Simplemkl., *J. of Mach. Learning Research.* **9**(11) (2008).

Chapter 8

Fuzzy Sets as a Logic Canvas for Pattern Recognition

Witold Pedrycz¹ and Nick J. Pizzi²

¹*Department of Electrical and Computer Engineering
University of Alberta
Edmonton, Canada
pedrycz@ee.ualberta.ca*

²*InfoMagnetics Technologies Corporation
Winnipeg, Canada
pizzi@imt.ca*

Imprecision and uncertainty are hallmarks of many pattern recognition problems. Although often impervious to crisp reasoning methods, these problems may be open to approximate ones. For this reason, the field has become a fertile and active domain with which to exercise fuzzy set-based modes of reasoning. For example, fuzzy sets support and enhance the design of logic-driven learning methods by: (i) formalizing prior domain knowledge; (ii) promoting hierarchical and modular architectures that exploit the most appropriate level of information granulation; (iii) providing an intuitive mapping between information couched in imprecise linguistic terminology and a well-defined contextual space. In this chapter, the authors begin with a top-down approach. They elaborate on the fundamentals of pattern recognition and fuzzy sets by focusing on the methodological issues. Subsequently, they refer to some concepts pertaining to Granular Computing and information granules. This constitutes a broader context as fuzzy sets themselves can be viewed as one of the possible realizations of information granules. Finally, they discuss some representative architectures highlighting the role of fuzzy sets.

8.1. Introduction: Fuzzy sets and pattern recognition

Pattern recognition, with its underlying philosophy, fundamental methodology, and suite of advanced algorithms, has established itself as a mature, well developed, information technology. Pattern recognition is a heteroge-

neous area exploiting a diversity of processing paradigms such as probability, statistics, neural networks and fuzzy sets.

Fuzzy pattern recognition has emerged almost at the time fuzzy sets came into existence. One of the first papers authored by Bellman, Kalaba, and Zadeh [1] has succinctly highlighted the key aspects of the technology of fuzzy sets being cast in the setting of pattern recognition. Since then we have witnessed many developments including a number of comprehensive review studies and books [2, 3]. The recent years saw a plethora of studies in all areas of pattern recognition including methodology, algorithms, and case studies [4–9]. Not attempting in any way to be exhaustive in the coverage of the area, we emphasize the main trends, as summarized below:

- Fuzzy sets contribute to pattern recognition as a vehicle formalizing any prior domain knowledge about the classification environment. This facilitates decisions about the classifiers structure, enhances learning methods (by avoiding cumbersome learning from scratch), and supports the design of logic-driven, transparent classifier architectures that are easier to train and easier to comprehend. In other words, fuzzy sets act a conceptual canvas for logic-based and designer/user-oriented classifiers.
- Fuzzy sets promote a notion of partial membership to a class. This is in accordance with many classification tasks where class assignment may not be binary.
- By emphasizing the role of information granulation, fuzzy sets promote a hierarchical and modular approach to pattern recognition, which becomes indispensable when handling complex classification problems in a modular fashion. In particular, this manifests itself in the form of hierarchical architectures with a clearly delineated knowledge-based layer and gives rise to the hierarchy of low-end, more specialized local classifiers located at the lower end. The role of the upper layer of the overall hierarchy (which is implemented with the aid of fuzzy sets and fuzzy logic) is to schedule activities of the classifiers at the lower end and summarize the findings there and eventually repair some discrepancies. The modularization of the classifier has strong evidence in practice. In general, one may develop individual, local classifiers that cope with regions in the feature space that are different as to the separability level of the patterns belonging to different classes. We have to deal with regions of the feature space where classes are well separated as well

as regions where the classes overlap. In the first instance, a linear classifier may be a good option. On the other hand, a nearest neighbor classifier is a good option for the regions where the classes overlap.

- Traditionally, pattern recognition techniques are divided into two main groups: supervised and unsupervised classification schemes. While generally useful, this simplified taxonomy is often not in rapport with real-world classification scenarios. Quite often we may encounter situations that fall in-between these two fundamental scenarios. For instance, there could be a mixture of labeled and unlabeled patterns. This calls for unsupervised learning (clustering) that comes with a certain level of partial supervision.

The breadth of investigations carried out at the junction of pattern recognition and fuzzy sets is highly visible. The number of publications in this area demonstrates its steady growth as visualized in Table 8.1.

Table 8.1. Statistics of publications in the area of pattern recognition and fuzzy sets obtained from Google Scholar (as of September 13, 2014).

Year	Number of Publications
1975	392
1980	600
1985	899
1990	2,390
1995	4,940
2000	8,770
2005	15,200
2010	21,700
2013	25,600

The objective of this study is to investigate the role of fuzzy sets in pattern recognition along the main points highlighted above and serves as an introduction to the area with several main objectives: We intend to demonstrate the main aspects of the technology of fuzzy sets (or Granular Computing) in the context of the paradigm of pattern recognition. In this regard, it is of interest to revisit the fundamentals and see how fuzzy sets enhance them at the conceptual, methodological and algorithmic level and what area of applications could benefit the most from the incorporation of the technology of fuzzy sets. We present several methodological aspects and present several selected detailed architectures and algorithms, which

demonstrate how the technology of fuzzy sets is used in pattern recognition.

In the overall presentation, we proceed with a top-down approach. We elaborate on the fundamentals of pattern recognition and fuzzy sets by focusing on the methodological issues. Within this context, we also refer to some ideas of Granular Computing and information granules. This constitutes a broader context as fuzzy sets themselves can be sought as one of the visible realizations of information granules. In the sequel, we discuss some representative architectures highlighting the role of fuzzy sets.

It is assumed that the reader is familiar with the basic ideas of pattern recognition and fuzzy sets; one may consult a number of authoritative references in these areas [10–15]. As a matter of fact, one can position the study in a more comprehensive settings of Granular Computing [16, 17] in which fuzzy sets are just instances of information granules. A number of results in fuzzy pattern recognition can be extended to the granular pattern recognition paradigm; throughout the chapter, we will be making some pertinent observations with this regard.

8.2. The methodology of fuzzy sets in pattern recognition

The concept of fuzzy sets augments the principles of pattern recognition in several ways. The well-established techniques are revisited and their conceptual and algorithmic aspects are extended. Let us briefly highlight the main arguments, which also trigger some intensive research pursuits, and exhibit some far-reaching consequences from an applied perspective.

The leitmotif is that fuzzy sets help realize user-centricity of pattern recognition schemes. Fuzzy sets are information granules of well-defined semantics that form a vocabulary of basic conceptual entities when problems are being formalized, models built, and decisions articulated. By expressing a certain suitable point of view of the problem at hand and promoting a certain level of specificity, fuzzy sets form an effective conceptual framework for pattern recognition. There are two essential facets of the overall aspects of the nature of user-centricity:

Class memberships are described by membership grades. This quantification is of interest as there could be patterns whose allocation to classes might not be completely described in a Boolean (yes-no) manner. The user is more comfortable talking about levels of membership of particular patterns. It is also more instrumental to generate classification results when presented with intermediate values of membership values. There is an associated flagging effect: in the case of two-class problems, member-

ship values in the vicinity of 0.5 are indicative of further need to analyze the classification results or engage some other classifiers to either gather evidence in favour of belongingness to the given class (which is quantified through higher values of the membership functions) or collect evidence which justifies a reduction of such membership degrees.

Fuzzy sets contribute to the specialized, user-centric feature space. The original feature space is transformed via fuzzy sets and produces a new feature space that is easier to understand and enhances higher effectiveness of the classifiers formed at the next phase. Similarly, through the use of fuzzy sets one could achieve a reduction of dimensionality of the original feature space. The nonlinearity effect introduced by fuzzy sets could be instrumental in reducing learning time of classifiers and enhancing their discriminative properties as well as improving their robustness. The tangible advantage results from the nonlinear character of membership functions. A properly adjusted nonlinearity could move apart patterns belonging to different classes and bring closer (contracts) those regions in which the patterns belong to the same category. Consider a membership function defined in $[0,1]$ (which is a one-dimensional feature space in which patterns to be classified are located) and governed by some nonlinear function $A(x)$. The ratio

$$\gamma = \frac{|A(x) - A(y)|}{|x - y|} \quad (8.1)$$

quantifies an effect of expansion or contraction of the feature space. We select a suitable membership function and estimate its parameters in such a way that the above index attains values lower than 1 if the patterns described by x and y belong to the same class (contraction) or exceeds 1 if the patterns belong to different classes and as such need to be made as far apart as possible.

The key facets of fuzzy set-based user-centricity might be looked at together in a sense of an overall interface layer of the core computing faculties of pattern recognition as illustrated in Fig. 8.1.

Fuzzy pattern recognition dwells on the concepts of information granules and exploits their underlying formalism. Information granules giving rise to the general idea of Granular Computing help offer a sound level of abstraction needed to address the issue of complexity when reducing the level of detail pattern recognition tasks are exposed to.

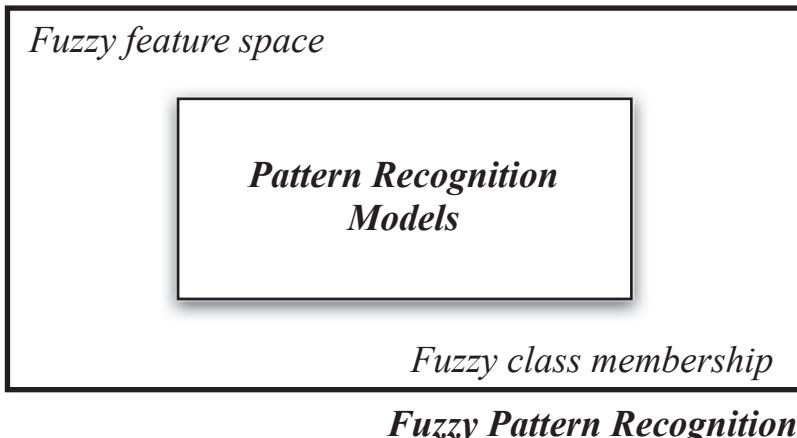


Fig. 8.1. Fuzzy sets forming an interface layer (feature space and class membership) and wrapping the core computational faculties of pattern recognition.

8.3. Information granularity and granular computing

Information granules permeate numerous human endeavors [16, 18]. No matter what problem is taken into consideration, we usually express it in a certain conceptual framework of basic entities, which we regard to be of relevance to the formulation and solution of a problem. This becomes a framework in which we formulate generic concepts adhering to some level of abstraction, carry out processing, and communicate the results to an external environment. Consider, for instance, image processing. In spite of the continuous progress in the area, a human being assumes a dominant and very much uncontested position when it comes to understanding and interpreting images. Surely, we do not focus our attention on individual pixels and process them as such but group them together into semantically meaningful constructs familiar objects we deal with in everyday life. Such objects involve regions that consist of pixels or categories of pixels drawn together because of their proximity in the image, similar texture, color, etc. This remarkable and unchallenged ability of humans dwells on our effortless ability to construct information granules, manipulate them and arrive at sound conclusions. As another example, consider a collection of time series. From our perspective, we can describe them in a semi-qualitative manner by pointing at specific regions of such signals. Specialists can effortlessly interpret ECG signals. They distinguish some segments of such

signals and interpret their combinations. Experts can interpret temporal readings of sensors and assess the status of the monitored system. Again, in all these situations, the individual samples of the signals are not the focal point of the analysis and the ensuing signal interpretation. We always granulate all phenomena (no matter if they are originally discrete or analog in their nature). Time is another important variable that is subjected to granulation. We use seconds, minutes, days, months, and years. Depending upon a specific problem we have in mind and who the user is, the size of information granules (time intervals) could vary quite dramatically. To the high-level management time intervals of quarters of year or a few years could be meaningful temporal information granules on basis of which one develops any predictive model. For those in charge of everyday operation of a dispatching plant, minutes and hours could form a viable scale of time granulation. For the designer of high-speed integrated circuits and digital systems, the temporal information granules concern nanoseconds, microseconds, and perhaps microseconds. Even such commonly encountered and simple examples are convincing enough to lead us to ascertain that (a) information granules are the key components of knowledge representation and processing, (b) the level of granularity of information granules (their size, to be more descriptive) becomes crucial to the problem description and an overall strategy of problem solving, (c) there is no universal level of granularity of information; the size of granules is problem-oriented and user dependent.

What has been said so far touched a qualitative aspect of the problem. The challenge is to develop a computing framework within which all these representation and processing endeavors can be formally realized. The common platform emerging within this context comes under the name of Granular Computing. In essence, it is an emerging paradigm of information processing. While we have already noticed a number of important conceptual and computational constructs built in the domain of system modelling, machine learning, image processing, pattern recognition, and data compression in which various abstractions (and ensuing information granules) came into existence, Granular Computing is innovative and intellectually proactive in several fundamental ways:

- It identifies the essential commonalities between the surprisingly diversified problems and technologies used which could be cast into a granular world (unified framework). This is a fully operational processing entity that interacts with the external world (that could be

another granular or numeric world) by collecting necessary granular information and returning the outcomes of the granular computing.

- With the emergence of the unified framework of granular processing, we get a better grasp as to the role of interaction between various formalisms and visualize a way in which they communicate.
- It combines the existing formalisms of set theory (interval analysis) [19], fuzzy sets [20–22], and rough sets [23–25] and clearly visualizes that in spite of distinct underpinnings and ensuing processing, they exhibit some fundamental commonalities. In this sense, Granular Computing establishes a stimulating environment of synergy between the individual approaches.
- By building upon commonalities of the existing formal approaches, Granular Computing helps build heterogeneous and multifaceted models of processing of information granules by clearly recognizing the orthogonal nature of some of the existing and well established frameworks (say, probability theory coming with its probability density functions and fuzzy sets with their membership functions).
- Granular Computing fully acknowledges a notion of variable granularity whose range could cover detailed numeric entities and very abstract and general information granules. It looks at the aspects of compatibility of such information granules and ensuing communication mechanisms of the granular worlds.
- Interestingly, the inception of information granules is highly motivated. We do not form information granules without reason. Information granules arise as an evident realization of the fundamental paradigm of abstraction.

Granular Computing forms a unified conceptual and computing platform. Yet, it directly benefits from the already existing and well-established concepts of information granules formed in the setting of set theory, fuzzy sets, rough sets and others. In the setting of this study it comes as a technology contributing to pattern recognition.

8.4. The algorithmic aspects of fuzzy set technology in pattern recognition: pattern classifiers

Each of these challenges comes with a suite of their own specific problems that require careful attention both at the conceptual as well as algorithmic

level. We have highlighted the list of challenges and in the remainder of this study present some of the possible formulations of the associated problems and look at their solutions. It is needless to say that our proposal points at some direction that deems to be of relevance however does not pretend to offer a complete solution to the problem. Some algorithmic pursuits are also presented as an illustration of some possibilities emerging there.

Indisputably, geometry of patterns belonging to different classes is a focal point implying an overall selection and design of pattern classifiers. Each classifier comes with its geometry and this predominantly determines its capabilities. While linear classifiers (built on a basis of some hyperplanes) and nonlinear classifiers (such as neural networks) are two popular alternatives, there is another point of view at the development of the classifiers that dwells on the concept of information granules. Patterns belonging to the same class form information granules in the feature space. A description of geometry of these information granules is our ultimate goal when designing effective classifiers.

8.4.1. Representatives of supervised fuzzy classifiers: fuzzy linear classifiers and fuzzy nearest neighbor classifiers

Linear classifiers [11] are governed by the well-known linear relationship

$$y(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0, \quad (8.2)$$

where w and w_0 are the parameters (weights and bias) of the classifier. The classification rule in the case of two classes (w_1 and w_2) reads as follows: classify \mathbf{x} to w_1 if $y(\mathbf{x}) > 0$ and assign to w_2 otherwise. The design of such a classifier (perceptron) has been intensively discussed in the literature and has resulted in a wealth of algorithms. The property of linear separability of patterns assures us that the learning method converges in a finite number of iterations. The classification rule does not quantify how close the pattern is to the linear boundary, which could be regarded as a certain drawback of this classifier. The analogue of the linear classifier expressed in the language of fuzzy sets brings about a collection of classifier parameters represented as fuzzy numbers, and triangular fuzzy numbers, in particular. The underlying formula of the fuzzy classifier comes in the form

$$Y(\mathbf{x}) = W_1 \otimes x_1 \oplus W_2 \otimes x_2 \dots W_n \otimes x_n \oplus W_0, \quad (8.3)$$

where W_i ($i = 1, 2, \dots, n$) are triangular fuzzy numbers. As a result, the classifier output is also a fuzzy number. Note that we used the symbols of

addition of multiplication to underline the fact that the computing is concerned with fuzzy numbers rather than plain numeric entities. A triangular fuzzy number, A , can be represented as a triple $A = \langle a_-, a, a_+ \rangle$ with “ a ” being the modal value of A and a_- and a_+ standing for the bounds of the membership function. The design criterion considered here stresses the separability of the two classes in the sense of the membership degrees produced for a given pattern, \mathbf{x} . More specifically, we have the following requirements

$$\max Y(\mathbf{x}), \text{ if } \mathbf{x} \in w_1, \min Y(\mathbf{x}), \text{ if } \mathbf{x} \in w_2. \quad (8.4)$$

Similarly, as commonly encountered in fuzzy regression, one could consider a crux of the design based on linear programming. In contrast to linear classifiers, fuzzy linear classifiers produce classification results with class quantification; so rather than a binary decision being generated, we come up with the degree of membership of a pattern to class w_1 . An illustration of the concept is illustrated in Fig. 8.2.

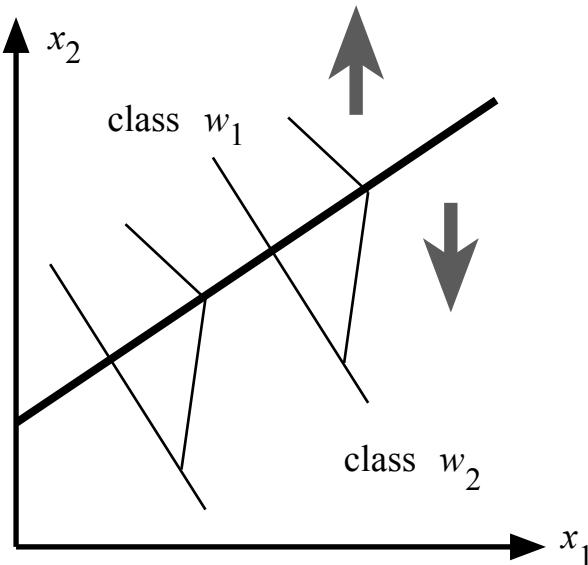


Fig. 8.2. Fuzzy linear classifier; note asymmetric nature of membership degrees generated around the classification boundary.

In virtue of the classification rule, the membership of $Y(\mathbf{x})$ is highly asymmetric. The slope at one side of the classification line is reflective

of the distribution of patterns belonging to class w_1 . The geometry of the classifier is still associated with a linear boundary. What fuzzy sets offer is a fuzzy set of membership associated with this boundary. Linear separability is an idealization of the classification problem. In reality there could be some patterns located in the boundary region that do not satisfy the linearity assumption. So even though the linear classifier is a viable first alternative, further refinement is required. The nearest neighbor (NN) classifier could form a sound enhancement of the fuzzy linear classifier. The popularity of NN classifiers stems from the fact that in their development we rely on lazy learning so no optimization effort is required at all. Any new pattern is assigned to the same class as its closest neighbor. The underlying classification rule reads as follows: given \mathbf{x} , determine \mathbf{x}_{i_0} in the training set such that $i_0 = \arg_i \min \|\mathbf{x} - \mathbf{x}_i\|$ assuming that the membership of \mathbf{x}_{i_0} is w_2 , \mathbf{x} is classified as w_2 as well.

The NN neighbor classifier could involve a single closest neighbor (in which case the classification rule is referred to as 1-NN classifier), 3 nearest neighbors, 5 nearest neighbors, or k (k is odd) nearest neighbors (referred to as a k -NN classifier). The majority vote implies the class membership of x . The extension of the k -NN classification rule can be realized in many ways. An intuitive approach is to compute a degree of membership of \mathbf{x} to a particular class by looking at the closest k neighbors, determining the membership degrees $u_i(x)$ ($i = 1, 2, \dots, L$), and choosing the highest one as reflective of the allocation of x to the that particular class (see Fig. 8.3); here $\text{Card}(G) = L$. Specifically, we have

$$u_i(\mathbf{x}) = \frac{1}{\sum_{x_j \in \Gamma}^L \left(\frac{\|\mathbf{x} - \mathbf{x}_i\|}{\|\mathbf{x} - \mathbf{x}_j\|} \right)^2}. \quad (8.5)$$

(We note a resemblance of this expression to the one describing membership degrees computed in the fuzzy c-means (FCM) algorithm [10]). Given that pattern i_0 where $i_0 = \max_{i=1,2,\dots,L} u_i(x)$ belongs to class w_1 , we assign x to the same class with the corresponding membership degree. The patterns positioned close to the boundary of the linear classifier are engaged in the NN classification rule.

The architecture illustrated in Fig. 8.4 is an aggregate of the fuzzy linear classifier and the fuzzy NN classifier that modifies the original membership degrees coming from the linear classifier by adjusting them on the basis of some local characteristics of the data. The NN classifier output (generating the highest membership degree) is governed by the expression $m = \max_{i=1,2} u_i(\mathbf{x})$ and $i_0 = \arg_i \max_{i=1,2} u_i(\mathbf{x})$. Let us now introduce

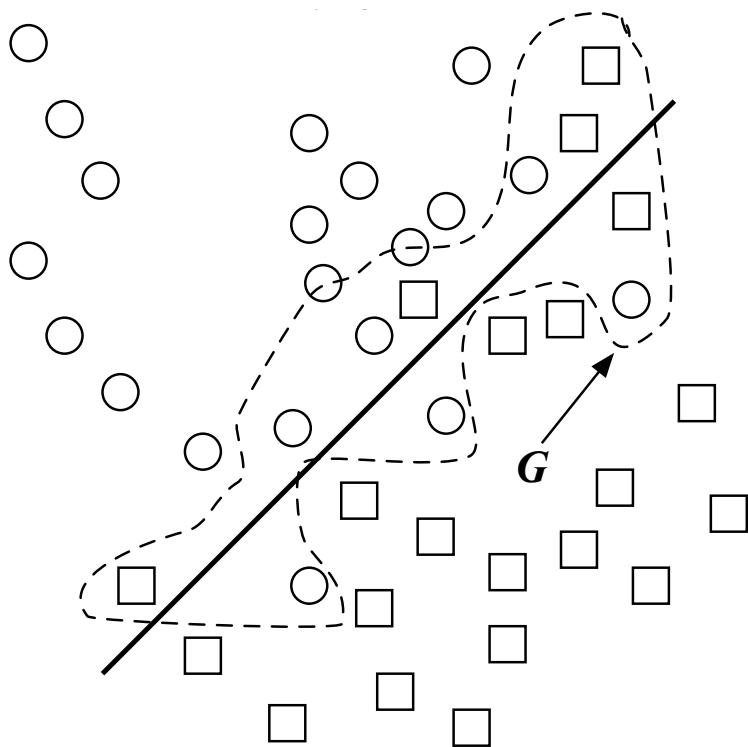


Fig. 8.3. Linear classifier with a collection of patterns in the boundary region (G) whose treatment is handled by NN classifier.

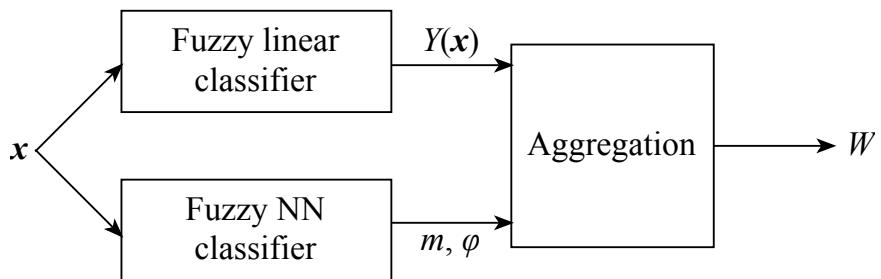


Fig. 8.4. Combination of two fuzzy classifiers: fuzzy linear classifier focused on the global nature of classification is adjusted by the results formed by the fuzzy NN classifier acting on a local basis.

the following indicator (or characteristic) function, φ

$$\varphi(\mathbf{x}) = \begin{cases} 1 & \text{if } i_0 = 1 \text{ (class } w_1), \\ 0 & \text{if } i_0 = 2 \text{ (class } w_2). \end{cases} \quad (8.6)$$

The aggregation of the classification results produced by the two classifiers is completed in the following fashion with the result W being a degree of membership to class w_1 ,

$$W = \begin{cases} \max(1, Y(\mathbf{x}) + m\varphi(\mathbf{x})) & \text{if } \varphi(\mathbf{x}) = 1, \\ \min(0, Y(\mathbf{x}) - m(1 - \varphi(\mathbf{x}))) & \text{if } \varphi(\mathbf{x}) = 0. \end{cases} \quad (8.7)$$

Note that if the nearest neighbor classifier has assigned pattern \mathbf{x} to class, w_1 , this class membership elevates the membership degree produced by the fuzzy linear classifier, hence we arrive at the clipped sum $\max(1, Y(\mathbf{x}) + m)$. In the opposite case, where the NN classification points at the assignment to the second class, w_1 , the overall class membership to w_1 becomes reduced by m . As a result, given a subset of patterns \mathbf{G} in the boundary region of the fuzzy linear classifier, the classification region is adjusted accordingly. Its overall geometry is more complicated and nonlinear adjusting the original linear form to the patterns located in this boundary region.

8.4.2. Fuzzy set based transparent topologies of the pattern classifier

Neural networks [26] occupy a significant niche in pattern recognition mainly in the capacity of nonlinear classifiers. While capable of carrying out various learning schemes, neural networks are inherently “black-box” architectures. This has two key implications: meaningful interpretations of the network’s structure are difficult to derive; any prior knowledge we may possess about the problem, which would reduce the total learning effort, is difficult to “download” onto the network (the common practice is to “learn from scratch”). A solution would be to construct networks that are similar to neural networks in terms of their learning abilities while at the same time composed of easily interpretable processing elements. To this end, a logic-oriented processing style is an asset. Bearing this in mind, [27] proposed two general classes of fuzzy AND and OR neurons. They serve as generalizations of standard AND and OR digital gates. The AND neuron is a static n -input single output processing element $y = \text{AND}(\mathbf{x}; \mathbf{w})$ constructed with fuzzy set operators (t-norms, t, and t-conorms, s),

$$y = \prod_{i=1}^n (x_i s w_i). \quad (8.8)$$

Here $\mathbf{w} = (w_1, w_2, \dots, w_n)^T$ denotes a neurons connections (weight vector). In light of triangular norm boundary conditions, we obtain

- If $w_i = 1$, then the corresponding input has no impact on the output. Moreover, the monotonicity property holds: higher values of w_i reduce the impact of x_i on the output of the neuron.
- If the elements of w assume values equal to 0 or 1, then the AND neuron becomes a standard AND gate. The OR neuron, denoted as $y = \text{OR}(\mathbf{x}; \mathbf{w})$ is described in the form

$$y = \sum_{i=1}^n (x_i t w_i). \quad (8.9)$$

As with AND neurons, the same general properties hold; the boundary conditions are somewhat complementary: the higher the connection value, the more evident the impact of the associated input on the output.

The fundamental Shannon's expansion theorem [28] states that any Boolean function can be represented as a sum of minterms (or equivalently, a product of maxterms). The realization is a two-layer digital network: the first layer has AND gates (realizing the required minterms); the second consists of OR gates that carry out OR-operations on the minterms. Fuzzy neurons operate in an environment of continuous variables. An analogy of the Shannon theorem (and the resulting topology of the network) can be realized in the form illustrated in Fig. 8.5. Here, AND neurons form a series of generalized minterms. The OR neurons serve as generalized maxterms. This network approximates experimental continuous data in a logic-oriented manner. In contrast, note that the sum of minterms represents Boolean data. The connections of the neurons equip the network with the required parametric flexibility. Alluding to the nature of approximation accomplished here, we will refer to the network as a logic processor (LP). The logic-based nature of the LP is crucial to a broad class of pattern classifiers. Figure 8.6 portrays a common situation encountered in pattern classification. Patterns belonging to the same class (say, w) occupy several groups (regions). For the sake of discussion, we consider only two features over which we define several fuzzy sets (granular landmarks) and through which we "sense" the patterns. The AND neurons build (combine) the fuzzy sets defined for individual variables. Structurally, these neurons develop fuzzy relations, regions where the patterns are concentrated. Then all these regions are aggregated using a single OR neuron. The connections help capture the details of the regions (AND neuron) and contribution of each region to the classification decision (OR neuron). In this setting, one

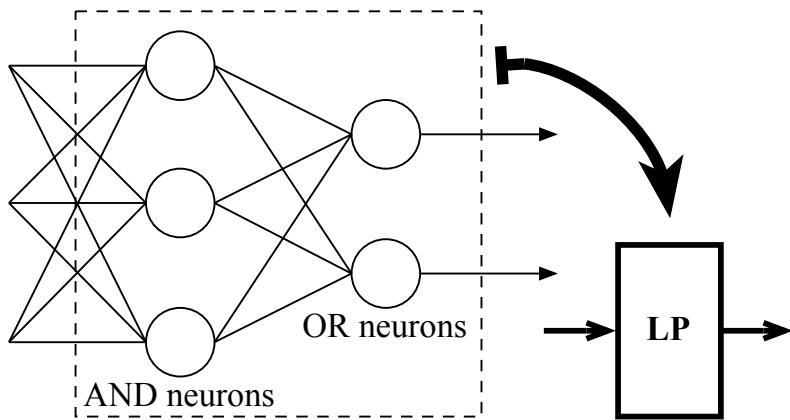


Fig. 8.5. Logic processor (LP): a general topology; the inputs involve both direct and complemented inputs.

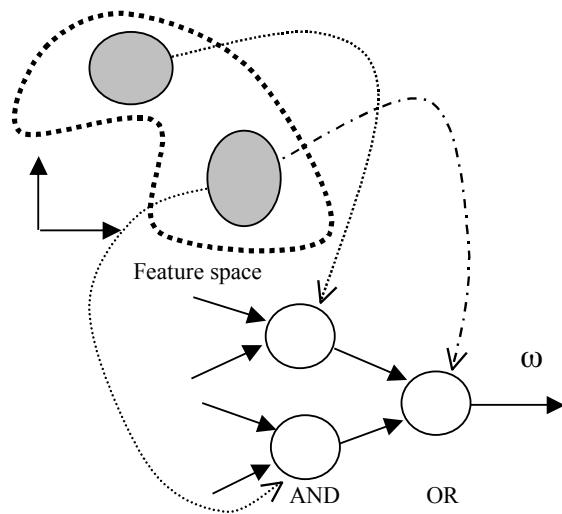


Fig. 8.6. LP as a fundamental logic-driven classification structure.

may refer to radial basis function (RBF) neural networks that resemble the above logic structure; however, the latter does not carry any strong logic connotation (see Table 8.2).

Table 8.2. Logic processor and RBF neural networks: a comparative analysis.

Logic Processor	RBF Neural Network
AND neurons: form the fuzzy relations of features in the feature space. Identify separate concentrations of pattern regions belonging to a given class. Learn neuron connections in order to model the details of these regions.	RBF (receptive fields) are used to identify homogeneous regions in the feature space in terms of patterns belonging to the class of interest. RBF may assume various functional forms (e.g., Gaussians) and may be adjusted by modifying its parameters (usually modal value and spread).
OR neurons: summarize the separate concentrations of pattern regions. Use connections to model impact of pattern regions on final classification decision. Some regions (outputs of AND neurons) need to be discounted, especially if they are heterogeneous and include patterns belonging to other classes.	A linear summation: the outputs of the RBFs (activation levels of these fields) are processed through a weighted sum where the weights are used to cope with the impact of the receptive fields on the final classification outcome.

8.4.3. *Granular constructs of a classifier*

The fundamental development strategy pursued when dealing with this classifier category dwells upon the synergistic and orchestrated usage of two fundamental technologies of Granular Computing, namely intervals (hyper-boxes) and fuzzy sets. Given this, the resulting constructs will be referred to as granular hyperbox-driven classifiers (HDC). The HDC architecture comes with two well-delineated architectural components that directly imply its functionality. The core (primary) part of the classifier, which captures the essence of the structure, is realized in terms of interval analysis. Sets are the basic constructs that form the feature space regions where there is high pattern homogeneity (which implies low classification error). We may refer to it as a core structure. Fuzzy sets are used to cope with patterns outside the core structure and in this way contribute to a refinement of the already developed core structure. This more detailed structure type will be referred to as secondary. The two-level granular architecture of the classifier reflects a way in which classification processes are usually carried out: we start with a core structure where the error is practically absent and then consider the regions of high overlap between classes where there is a high likelihood of error. For the core structure, the use of sets as generic information granules is highly legitimate: there is no need to distinguish between these elements of the feature space. The areas of high overlap require more detailed treatment hence here arises a genuine need

to consider fuzzy sets as suitable granular constructs. The membership grades play an essential role in expressing confidence levels associated with the result. In this way, we bring detailed insight into the geometry of the classification problem and identify regions of poor classification. One can view the granular classifier as a two-level hierarchical classification scheme whose development adheres to a stepwise refinement of the construct with sets forming the architecture core and fuzzy sets forming its specialized enhancement. Given this, a schematic view of the two-level construct of the granular classifier is shown in Fig. 8.7. One of the first approaches to the construction of set-based classifiers (hyperboxes) was presented by Simpson [29, 30] both in supervised and unsupervised mode. Abe *et al.* [31] presented an efficient method for extracting rules directly from a series of activation hyperboxes, which capture the existence region of data for a given class and inhibition hyperboxes, which inhibit the existence of data of that class. Rizzi *et al.* [32, 33] proposed an adaptive resolution classifier (ARC) and its pruned version (PARC) in order to enhance the constructs introduced by Simpson. ARC/PARC generates a regularized min-max network by a series of hyperbox cuts. Gabrys and Bargiela [34] described a general fuzzy min-max (GFMM) neural network that combines mechanisms of supervised and unsupervised learning into a single unified framework.

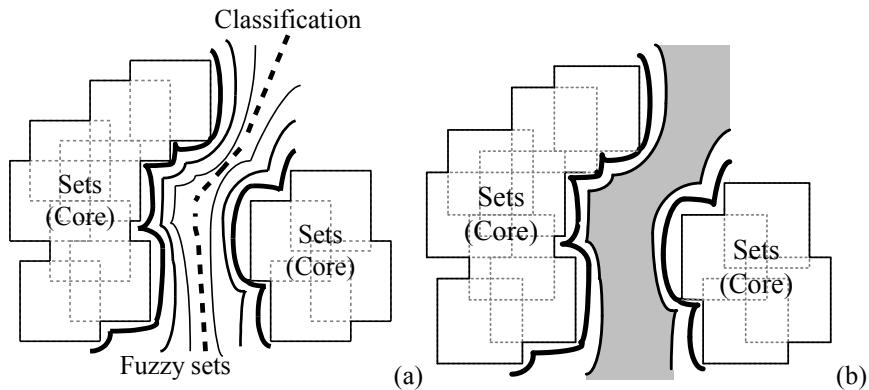


Fig. 8.7. From sets to fuzzy sets: a principle of a two-level granular classifier exploiting the successive usage of the information granulation formalisms (a), and further refinements of the information granules realized on a basis of the membership degrees (b).

The design of the granular classifiers offers several advantages over some “standard” pattern classifiers. First, the interpretability is highly enhanced:

both the structure and the conceptual organization appeals in a way in which an interpretation of the topology of patterns is carried out. Second, one can resort to the existing learning schemes developed both for set-theoretic classifiers and fuzzy classifiers [35, 36].

8.5. Fuzzy generalization of the Petri net

Let us briefly recall the basic concept of a Petri net. A Petri net is a finite graph with two types of nodes, known as places, P , and transitions, T . More formally, the net can be viewed as a triplet (P, T, F) composed of places, transitions and a flow relation, F , showing links among places and transitions.

The elements of F are the arcs of the Petri net. Each place comes equipped with some tokens that form a marking of the Petri net. The flow of tokens in the net occurs through firings of the transitions; once all input places of a given transition have a nonzero number of tokens, this transition fires. Subsequently, the tokens are allocated to the output places of the transition. Simultaneously, the number of tokens at the input places is reduced. The effect of firing the transitions is binary: the transition either fires or does not fire.

An important generalization of the generic model of the Petri net is to relax the Boolean character of the firing process of the transition [37]. Instead of subscribing to the firing-no firing dichotomy, we propose to view the firing mechanism as a gradual process with a continuum of possible numeric values of the firing strength (intensity) of a given transition. Subsequently, the flow of tokens can also take this continuum into consideration. Evidently, such a model is in rapport with a broad class of real-world phenomena including pattern classification. The generalization of the net along this line calls for a series of pertinent realization details. In what follows, we propose a construct whose functioning adheres as much as possible to the logic fabric delivered by fuzzy sets. In this case, a sound solution is to adopt the ideas of fuzzy logic as the most direct way of implementation of such networks.

8.5.1. The architecture of the fuzzy Petri net

Cast in the framework of pattern classification, the topology of the fuzzy Petri net is portrayed in Fig. 8.8. As will be shown further on, this setting correlates well with the classification activities encountered in any process

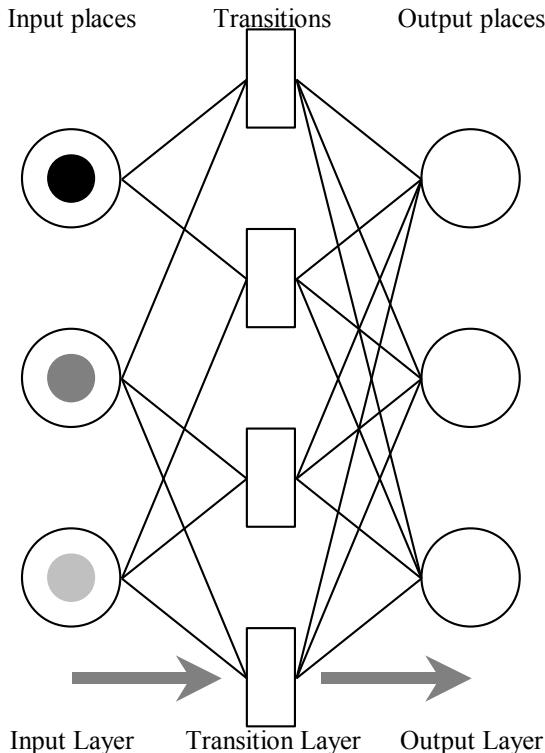


Fig. 8.8. A general three-layer topology of the fuzzy Petri net.

of pattern recognition. The network constructed in this manner comprises three layers:

- An input layer composed of n input places;
- A transition layer composed of hidden transitions;
- An output layer consisting of m output places.

The input place is marked by the value of the feature (we assume that the range of the values of each feature is in the unit interval). These marking levels are processed by the transitions of the network whose levels of firing depend on the parameters associated with each transition such as their threshold values and the weights (connections) of the incoming features. Subsequently, each output place corresponds to a class of patterns distinguished in the problem. The marking of this output place reflects a

level of membership of the pattern in the corresponding class. The detailed formulas of the transitions and output places rely on the logic operations encountered in the theory of fuzzy sets. The i^{th} transition (more precisely, its activation level z_i) is governed by the expression

$$z_i = \frac{n}{j=1} [w_{ij} s(r_{ij} \rightarrow x_j)], \quad (8.10)$$

where:

- w_{ij} is a weight (connection) between transition i and input place j ;
- r_{ij} is a marking level threshold for input place j and transition i ;
- x_j is the marking level of input place j ;
- T denotes a t-norm and s denotes an s-norm.

The implication operator, \rightarrow , is expressed in the form

$$a \rightarrow b = \sup \{c \in [0, 1] \mid \text{at } c \leq b\}, \quad (8.11)$$

where a and b are the arguments of the implication operator confined to the unit interval. Note that the implication is induced by a certain t-norm. In the case of two-valued logic, (8.11) returns the same truth values as the commonly-known implication operator, namely,

$$a \rightarrow b = \begin{cases} b & \text{if } a > b \\ 1 & \text{otherwise} \end{cases} = \begin{cases} 0 & \text{if } a = 1, b = 0 \\ 1 & \text{otherwise} \end{cases} \quad (a, b \in 0, 1). \quad (8.12)$$

Output place j (more precisely, its marking y_j) summarizes the levels of evidence produced by the transition layer and performs a nonlinear mapping of the weighted sum of the activation levels of these transitions, z_i , and the associated connections v_{ji} ,

$$y_j = f \left(\sum_{i=1}^{\text{hidden}} v_{ji} z_i \right) \quad (8.13)$$

where f is a nonlinear monotonically increasing mapping from $[0, 1]$ to $[0, 1]$.

The role of the connections of the output places is to modulate the impact exhibited by the firing of the individual transitions on the accumulation of the tokens at the output places (namely, the membership value of the respective class). The negative values of the connections have an inhibitory effect, that is, the value of the class membership gets reduced. Owing to the type of aggregation operations used in the realization of the transitions, their interpretation sheds light on the way in which the individual features of the problem are treated. In essence, the transition produces

some higher level, synthetic features out of those originally encountered in the problem and represented in the form of the input places. The weight (connection) expresses a global contribution of feature j to transition i : the lower the value of W_{ij} , the more significant the contribution of the feature to the formation of the synthetic aggregate feature formed at the transition level. The connection itself is weighted uniformly regardless of the numeric values it assumes. The more selective (refined) aggregation mechanism is used when considering threshold values. Referring to (8.10), one finds that the thresholding operation returns 1 if X_j exceeds the value of the threshold r_{ij} . In other words, depending on this threshold level, the level of marking of the input place is “masked” and the threshold operation returns 1. For the lower values of the marking, such levels are processed by the implication operation and contribute to the overall level of transition firing.

One should emphasize that the generalization of the Petri net proposed here is in full agreement with the two-valued generic version of the net commonly encountered in the literature. Consider, for instance, a single transition (transition node). Let all its connections and thresholds be restricted to $\{0,1\}$. Similarly, the marking of the input places is also quantified in a binary way. Then the following observations are valid:

- Only those input places are relevant to the functioning of transition i for which the corresponding connection are set to 0 and whose thresholds are equal to 1. Denote a family of these places by \mathbf{P} .
- The firing level of transition i is described by,

$$z_i = \prod_{j \in \mathbf{P}}^n (r_{ij} \rightarrow x_j) \quad (8.14)$$

It becomes apparent that the firing level is equal to 1 if and only if the marking of all input places in \mathbf{P} assume 1; the above expression for the transition is an and-combination of the marking levels of the places in \mathbf{P} ,

$$z_i = \prod_{j \in \mathbf{P}}^n x_j. \quad (8.15)$$

(Let us recall that any t-norm can be used to model the and operation; moreover, all t-norms are equivalent when operating on the 0-1 truth-values). Fuzzy Petri nets deliver another resource-based insight into the way in which pattern classifiers function. The role of resources is assumed by the features. Input places relate to fuzzy sets defined in the feature space. The higher the marking (satisfaction) of the input places, the higher the transition level, which translates into the output place's firing level (class assignment).

8.5.2. The learning procedure

Learning in the fuzzy Petri net is parametric in nature, meaning that it focuses on changes (updates) of the parameters (connections and thresholds) of the net, its structure is unchanged. These updates are carried out in such a way that a predefined performance index is minimized. To concentrate on the detailed derivation of the learning formulas, it is advantageous to view a fully annotated portion of the network as illustrated in Fig. 8.9. The performance index to be minimized is viewed as a standard sum of squared errors. The errors are expressed as differences between the levels of marking of the output places of the network and their target values. The considered on-line learning assumes that the modifications to the parameters of the transitions and output places occur after presenting an individual pair of the training sets, say marking of the input places (denoted by x) and the target values (namely, the required marking of the output places) expressed by t . Then the performance index for the input-output pair is

$$Q = \sum_{k=1}^m (t_k - y_k)^2. \quad (8.16)$$

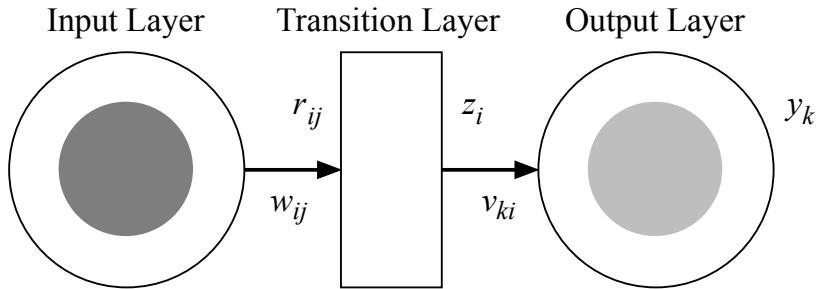


Fig. 8.9. Optimization in the fuzzy Petri net; a section of the net outlines all notation being used in the learning algorithm.

The standard gradient-based method governs the connection updates

$$\text{param}(\text{iter} + 1) = \text{param}(\text{iter}) - \alpha \nabla \text{param } Q, \quad (8.17)$$

where $\nabla \text{param } Q$ is a gradient of the performance index Q taken with respect to the parameters of the fuzzy Petri net. The iterative character of the learning scheme is underlined by the parameter vector regarded as a function of successive learning epochs. The learning intensity is controlled

by the positive learning rate, α . In the above scheme, the vector of parameters, **param**, is used to encapsulate the elements of the structure to be optimized. A complete description of the update mechanism will be described below. With the quadratic performance index (8.16) in mind, the following holds

$$\nabla_{\text{param}} Q = -2 \sum_{k=1}^m (t_k - y_k) \nabla_{\text{param}} y_k. \quad (8.18)$$

Refer to Fig. 8.9 for the following derivations. The nonlinear function associated with the output place (8.13) is a standard sigmoid nonlinearity

$$y_k = \frac{1}{1 + e^{-z_k}}. \quad (8.19)$$

For the connections of the output places we obtain

$$\frac{\delta y_k}{\delta v_{ki}} = y_k(1 - y_k)z_i, \quad (8.20)$$

where $k = 1, 2, \dots, m$ and $i = 1, 2, \dots$, hidden. Note that the derivative of the sigmoidal function is equal to $y_k(1 - y_k)$. Similarly, the updates of the threshold levels of the transitions of the net are expressed in the form

$$\frac{\delta y_k}{\delta r_{ij}} = \frac{\delta y_k}{\delta z_i} \frac{\delta z_i}{\delta r_{ij}} \quad (8.21)$$

where $i = 1, 2, \dots, n$. We then obtain

$$\frac{\delta y_k}{\delta z_i} = y_k(1 - y_k)v_{ki} \quad (8.22)$$

and

$$\begin{aligned} \frac{\delta z_i}{\delta r_{ij}} &= A \frac{\delta}{\delta r_{ij}} (w_{ij} + (r_{ij} \rightarrow x_j) - w_{ij}(r_{ij} \rightarrow x_j)) \\ &= A(1 - w_{ij}) \frac{\delta}{\delta r_{ij}} (r_{ij} \rightarrow x_j) \end{aligned} \quad (8.23)$$

where the new expression, denoted by A , is defined by taking the t-norm over all the arguments except j

$$A = \prod_{\substack{l=1 \\ l \neq j}}^n [w_{il} s(r_{il} \rightarrow x_l)] \quad (8.24)$$

The calculations of the derivative of the implication operation can be completed once we confine ourselves to some specific realization of the t-norm that is involved in its induction. For the product (being a particular example of the t-norm), the detailed result reads as

$$\frac{\delta}{\delta r_{ij}} (r_{ij} \rightarrow x_j) = \frac{\delta}{\delta r_{ij}} \begin{cases} \frac{x_j}{r_{ij}} & \text{if } r_{ij} > x_j, \\ 1 & \text{otherwise;} \end{cases} = \begin{cases} -\frac{x_j}{r_{ij}^2} & \text{if } r_{ij} > x_j \\ 0 & \text{otherwise.} \end{cases} \quad (8.25)$$

The derivatives of the transition nodes are obtained in a similar way

$$\frac{\delta y_k}{\delta w_{ij}} = \frac{\delta y_k}{\delta z_i} \frac{\delta z_i}{\delta w_{ij}}, \quad (8.26)$$

where $k = 1, 2, \dots, m$, $i = 1, 2, \dots$, hidden, and $j = 1, 2, \dots, n$. Subsequently

$$\frac{\delta z_i}{\delta w_{ij}} = A \frac{\delta}{\delta w_{ij}} (w_{ij} + (r_{ij} \rightarrow x_j) - w_{ij}(r_{ij} \rightarrow x_j)) = A(1 - (r_{ij} \rightarrow x_j)). \quad (8.27)$$

There are two aspects of further optimization of the fuzzy Petri nets that need to be raised in the context of their learning:

- *The number of transition layer nodes:* The optimization of the number of transition nodes of the fuzzy Petri net falls under the category of structural optimization that cannot be handled by the gradient-based mechanisms of the parametric learning. By increasing the number of these nodes, we enhance the mapping properties of the net, as each transition can be fine-tuned to fully reflect the boundaries between the classes. Too many of these transitions, however, could easily develop a memorization effect that is well known in neural networks.
- *The choice of t- and s-norms:* This leads us to a semi-parametric optimization of the fuzzy Petri net. The choice of these norms does not impact the architecture but in this optimization we cannot resort to gradient-based learning. A prudent strategy is to confine oneself to a family of t- and s-norms that can be systematically exploited.

Table 8.3 summarizes the main features of fuzzy Petri nets and contrast them with the structures the proposed constructs have in common, namely Petri nets and neural networks. Fuzzy Petri nets combine the advantages of both neural networks in terms of learning abilities with the glass box-style processing (and architectures) of Petri nets.

8.6. Supervised, unsupervised, and hybrid modes of learning

Supervised and unsupervised models of learning are commonly used taxonomies of pattern classifiers. Classifiers constructed in supervised learning mode exploit knowledge about membership of patterns to classes. In this way, the main task is to design a nonlinear mapping that discriminates between patterns belonging to different categories so that a classification error

Table 8.3. A comparative analysis of Fuzzy Petri nets, Petri nets, and neural networks.

	Learning	Knowledge Representation
Petri Nets	From nonexistent to significantly limited (depending on the type of Petri nets).	Transparent representation of knowledge arising from mapping a given problem (specification) onto the net structure. Well-defined semantics of transitions and places.
Fuzzy Petri Nets	Significant learning abilities (parametric optimization of the net connections). Structural optimization can be exercised through a variable number of the transitions utilized in the net.	Transparent representation of knowledge (glass box processing style). Problem (its specification) is mapped directly onto the topology. Fuzzy sets deliver an essential feature of continuity required to cope with continuous phenomena encountered in many problems.
Neural Networks	High learning abilities and a vast number of learning algorithms. Learning scheme properties are well known (advantages and disadvantages).	Limited (black box style of processing) mainly due to the highly distributed network topologies and homogeneous processing elements (neurons) used throughout the overall network. As a result, one has to provide additional interpretation mechanisms for these networks.

assumes a minimal value. The other criterion concerns an efficiency of learning/design of the classifier, which has to be high in case of large data sets. Obviously, a Bayesian classifier is a reference point for all constructs. Both neural networks and fuzzy set architectures along with their hybrids [38–41] are commonly exploited here. Unsupervised learning assumes that we have a collection of patterns that do not carry labels (class assignment) and these need to be “discovered” in the process of learning or self-organization. Unsupervised learning models include well-known neural network architectures such as self-organizing maps [42] and the family of ART algorithms [43]. Fuzzy sets offer a wide family of FCM clustering methods and have been successfully exploited in a number of different applications [13, 44–47]. In general, there are numerous clustering algorithms that differ in terms of their underlying optimization criteria, methods of handling data, and a format of results being produced (dendograms, partition matrices, etc.). In particular, we are interested in objective function-based clustering where

a structure in the given data set is determined (“discovered”) by minimizing a certain performance index (objective function). Our anticipation is that the minimized objective function relates with the “optimal” structure discovered in the analyzed data sets. The form of the objective function (Q) is usually given in advance and does not change during the optimization process. The results of clustering are represented in the form of a partition matrix, U . Partition matrices store details regarding class membership of the respective patterns (data). Each row of the matrix corresponds to the respective class (group) identified in the data set. Fuzzy clustering gives rise to the rows that contain membership functions of the fuzzy sets. In the case of clustering we get characteristic functions of sets determined in the entire data set.

While the above taxonomy is sound, the reality of classification problems calls for an array of learning modes. Consider a handful of examples making this point evident and justifying a continuum of models of hybrid supervised-unsupervised learning:

- A mixture of labeled-unlabeled patterns. While most of the patterns in the mixture are not labeled, there is a fraction of selected patterns with labels attached. These patterns can serve as “anchor” points guiding a formation of the overall structure in the data. The situations leading to such mixtures are specific to the problems where there is a substantial cost of classifying the patterns, the job of class assignment is tedious and, finally, the data set is huge. For instance, a collection of handwritten characters falls under this category. The size of the data set is evidently very large (as the characters are readily available). The labeling of all of them is not feasible. Yet a portion of the data set can be manually inspected and the pattern labeled.
- The labeling may not be fully reliable. This could be a result of errors made by the individual making class assignment, complexity of the classification problem, or a lack of sharp class boundaries. This yields patterns whose membership to some classes may be questionable and therefore need to be treated as such during classifier construction.

An example of a preprocessing strategy that compensates for the possible imprecision of class labels is *fuzzy class label adjustment* [48]: using training vectors, robust measures of location and dispersion are computed for each class center. Based on distances from these centers, fuzzy sets are

constructed that determine the degree to which each input vector belongs to each class. These membership values are then used to adjust class labels for the training vectors.

The aspect of partial supervision can manifest itself through various ways of utilizing domain knowledge about the problem or incorporating the preferences of the designer as to structure determination cast in a certain setting (context).

In what follows, we discuss two examples of clustering algorithms that fall under this category. The first one, called context-based clustering is concerned with clustering guided by some context variable (more precisely, a fuzzy set defined therein). The second one deals with clustering in the presence of some auxiliary pattern labeling (where we are concerned with a “reconciliation” of these labels with the structure in the data set itself).

As we will be working with fuzzy FCM method and view it as a starting point, it is instructive to set up necessary notation. For the data set X , consisting of N records, of dimensionality n , in which we are seeking c clusters (groups), the respective partition matrix is comprised of N columns and c rows. If we are interested in determining Boolean clusters, then the entries of U are 0-1 valuations. For fuzzy clusters, the partition matrix entries assume values in the unit interval; the ik^{th} entry u_{ik} denotes a degree of membership of pattern k to cluster i . In addition to the partition matrix, the clustering algorithm returns a set of prototypes: centroids describing the clusters in terms of single representatives. The optimization problem is governed by the objective function,

$$Q = \sum_{i=1}^c \sum_{k=1}^N u_{ik}^m \|x_k - v_i\|^2 \quad (8.28)$$

The partition matrix, U , satisfies a number of extra conditions; these properties are captured by restricting the optimization to the class of partition matrices U , say $U \in \mathbf{U}$ (FCM). The partition matrix, U , satisfies a number of extra conditions; these properties are captured by restricting the optimization to the class of partition matrices, U , say $U \in \mathbf{U}$,

$$\mathbf{U} = \left\{ u_{ik} \in [0, 1] \mid \sum_{i=1}^c u_{ik} = 1 \text{ and } 0 < \sum_{k=1}^N u_{ik} < N \right\} \quad (8.29)$$

where $i = 1, 2, \dots, c$. The other elements of the clustering method are:

- Prototypes (centroids) of the clusters are representatives of the determined groups of elements (data);

- The distance function, $\|\cdot\|$, describes the distance between the individual patterns and the prototypes;
- The fuzzification coefficient, $m > 1$, controls the level of fuzziness assigned to the clusters. By default, we assume $m = 2$.

In spite of differences resulting from the way the optimization task is formulated at the objective function level, the underlying idea remains the same: we intend to reveal a general structure in the data set.

8.6.1. *Context-based fuzzy clustering*

The essence of context-based clustering [49] is to search for structure within the data while making the search more focused by applying a context. More profoundly, our intent is to concentrate a search of structure with a certain variable (classification variable) in mind. One should stress that generic clustering is relation-driven, meaning that all variables play the same role in the overall design of the clusters. In the variable of interest, we distinguish a number of so-called contexts. The context itself is an information granule [18, 50] (captured by fuzzy sets) that is defined in one or several attributes whereby the search is focused for structure in the data. In other words, the formulation, “Reveal a structure in data X ”, is reformulated as, “Reveal a structure in data X in context A ”, where A denotes an information granule of interest (context of clustering). For instance, a task may be formulated as, “Reveal a structure in X for context equal to *high temperature*”, with *high temperature* being a fuzzy set defined in the context space. The context space (variable) alludes to the continuous classification variable.

Note that the selected information granule (context) directly impacts the resulting data to be examined. As a matter of fact, the context can be regarded as a window (or a focal point) of the clustering pursuit. On a technical side, the introduced linguistic context [50] provides us with a certain tagging of the data. Fig. 8.10 illustrates this phenomenon. The conditioning aspect (context sensitivity) of the clustering mechanism is introduced into the algorithm by taking into consideration the conditioning variable (context) assuming the values f_1, f_2, \dots, f_N on the corresponding patterns. More specifically, f_k describes a level of involvement of x_k in the assumed context, $f_k = A(x_k)$. The way in which f_k can be associated with or allocated among the computed membership values of x_k , say $u_{1k}, u_{2k}, \dots, u_{ck}$,

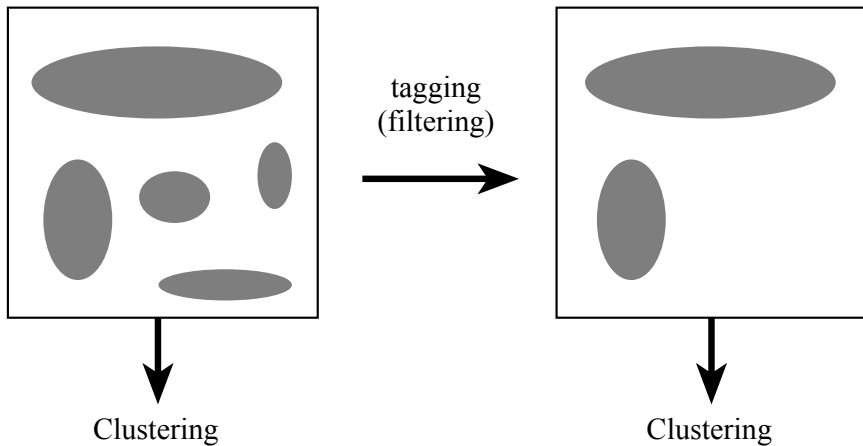


Fig. 8.10. An example of the tagging (logic filtering) effect provided by a linguistic context A ; note that some groups of data present in the original data set have been filtered out by the use of the fuzzy set in the context variable.

can be realized in the form,

$$\sum_{i=1}^c u_{ik} = f_k \quad (8.30)$$

that holds for all data points, $k = 1, 2, \dots, N$. The way in which the fuzzy set of context, A , is obtained will be discussed in a subsequent section. However, Fig. 8.11 highlights the way we proceed from data to clusters and the corresponding labeling by the membership values of the context variable. It is worth noting that the finite support of A “eliminates” some data points (for which the membership values are equal to zero) thus leaving only a certain subset of the original data to be used for further clustering. Bearing this in mind, we modify the requirements for the partition matrices and define the family,

$$\mathbf{U}(A) = \left\{ u_{ik} \in [0, 1] \mid \sum_{i=1}^c u_{ik} = f_k \forall k \text{ and } 0 < \sum_{k=1}^N u_{ik} < N \forall i \right\}. \quad (8.31)$$

Note that the standard normalization condition where the membership values sums up to 1 is replaced by the involvement (conditioning) constraint. The optimization problem may now be reformulated accordingly,

$$\min_{U, \nu_1, \nu_2, \dots, \nu_c} Q, \quad (8.32)$$

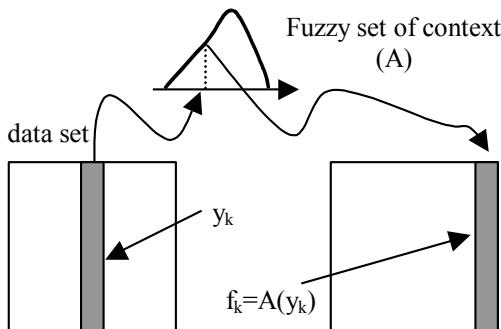


Fig. 8.11. From context fuzzy set to the labeling of data to be clustered (shadowed column in the data set consists of membership values of the context fuzzy set, A).

subject to

$$U \in \mathbf{U}(A). \quad (8.33)$$

Let us now proceed with deriving a complete solution to this optimization problem. Essentially, it may be divided into two separate subproblems:

- Optimization of the partition matrix \mathbf{U} ;
- Optimization of the prototypes.

As these two tasks may be handled independently of each other, we start with the partition matrix. Moreover, we notice that each column of U can be optimized independently, so let us fix the index of the data point, k . Rather than being subject to (8.33), (8.26) is now subject to the constraint,

$$\sum_{i=1}^c u_{ik} = f_k \quad (8.34)$$

In other words, having fixed the data index, we must solve N independent optimization problems. To make the notation more concise, we introduce the notation d_{ik} for the distance between the pattern and prototype,

$$d_{ik}^2 = \|\mathbf{x}_k - \mathbf{v}_i\|^2 \quad (8.35)$$

As the above is an example of optimization with constraints, we can easily convert this into unconstrained optimization by using the technique

of Lagrange multipliers. The overall algorithm is summarized as the following sequence of steps (where the fuzzification parameter ($m > 1$) affects the form of the membership functions describing the clusters):

Given: The number of clusters, c . Select the distance function, $\|\cdot\|$, termination criterion, $e > 0$, and initialize partition matrix $U \in \mathbf{U}$. Select the value of the fuzzification parameter, $m > 1$ (the default is $m = 2.0$).

- (1) Calculate centers (prototypes) of the clusters

$$\mathbf{v}_i = \frac{\sum_{k=1}^N u_{ik}^m \mathbf{x}_k}{\sum_{k=1}^N u_{ik}^m}, \quad (i = 1, 2, \dots, c)$$

- (2) Update partition matrix

$$u_{ik} = \frac{f_k}{\sum_{j=1}^c \frac{\|\mathbf{x}_k - \mathbf{v}_i\|^{\frac{2}{m-1}}}{\|\mathbf{x}_k - \mathbf{v}_j\|}}, \quad (i = 1, 2, \dots, c, j = 1, 2, \dots, N).$$

- (3) Compare U' to U . If termination criterion $\|U' - U\| < e$ is satisfied then stop, else return to step (1) with U equal to U' .

Result: partition matrix and prototypes.

8.6.2. Clustering with partial supervision: reconciling structural and labeling information

In the simplest case, we envision a situation where there is a set of labeled data (patterns) yet their labeling may not be perfect (as being completed by an imperfect teacher). Bearing this in mind, we want to confront this labeling with the “hidden” structure in the data and reconcile it with this inherent structure. Another interesting situation occurs when an expert provides a granulation of a concept or a variable formulated in terms of fuzzy sets or relations and we intend to reconcile it with the data. In any case, we anticipate that the linguistic labels should have enough experimental justification behind them and in this way make them sounder. This reconciliation is an example of building constructs on a basis of knowledge (here coming in the form of fuzzy sets) and an array of numeric data. Another version would involve labeling coming from some other clustering mechanisms and presenting results of clustering carried out for the data. The algorithmic aspects can be laid down in many ways. Incorporation of

the labeling information is reflected in the augmented objective function,

$$Q = \sum_{i=1}^c \sum_{k=1}^N u_{ik}^m \|x_k - v_i\|^2 + a \sum_{i=1}^c \sum_{k=1}^N (u_{ik} - f_{ik})^m b_k \|x_k - v_i\|^2. \quad (8.36)$$

The first term is the objective function, (8.28), used by the original FCM method; the second term reflects the domain knowledge introduced in the form of the partition matrix $F = [f_{ik}], i = 1, 2, \dots, c, k = 1, 2, \dots, N$.

The vector $\mathbf{b} = [b_k]$, which assumes values in the unit interval, helps to balance between the structural aspects coming from the clustering itself and the class membership coming from the external source. In addition, the weight set to 1, excludes the corresponding data point as not being externally labeled. Put differently: higher entries of \mathbf{b} stipulate a higher level of confidence associated with the labeling coming from the external source (say, expert). And conversely, lower entries of \mathbf{b} discount the labeling (class assignment) as not being reliable enough. The weight coefficient, a , plays a similar role as the previous vector but operates at the global level and does not distinguish between individual patterns.

8.7. Data and dimensionality reduction

The pattern recognition problem of dimensionality reduction [11] and complexity management, by no means a new endeavor, has led to intensively used classic techniques. There are approaches deeply rooted in classic statistical analysis such as principal component analysis and Fisher analysis that are of paramount relevance. What has changed quite profoundly over the decades is the magnitude of the problem itself that has forced us to explore new ideas and optimization techniques involving advanced techniques of global search including tabu search and biologically-inspired optimization mechanisms.

In a nutshell, we can distinguish between two fundamental reduction processes involving (a) data and (b) features (attributes). Data reduction is concerned with grouping patterns and revealing their structure in the form of clusters (groups). Clustering is regarded as one of the fundamental techniques within the domain of data reduction. Typically, we start with thousands of data points and arrive at 10-15 clusters. The nature of the clusters could vary depending upon the underlying formalisms. While in most cases, the representatives of the clusters are numeric entities such as prototypes or medoids, we can encounter granular constructs such as hyperboxes.

Feature or attribute reduction [12, 51–53] deals with: (a) transformation of the feature space into another feature space of a far lower dimensionality or (b) selection of a subset of features that are regarded to be the most essential (dominant) with respect to a certain predefined objective function. Considering the underlying techniques of feature transformation, we encounter a number of classic linear statistical techniques such as e.g., principal component analysis or more advanced nonlinear mapping mechanisms realized by, for example, neural networks.

The criteria used to assess the quality of the resulting (reduced) feature space give rise to two general categories, *filters* and *wrappers*. Using filters, we consider some criterion that pertains to the statistical *internal* characteristics of the selected attributes and evaluate them with this respect. In contrast, when dealing with wrappers, we are concerned with the effectiveness of the features to carry out classification, so in essence there is a mechanism (that is, a certain classifier) that effectively evaluates the performance of the selected features with respect to their discriminatory (that is *external*) capabilities. In addition to feature and data reduction being regarded as two separate processes, we may consider their combinations in which both features and data are reduced. The general view of the reduction mechanisms is presented in Fig. 8.12. Fuzzy sets augment the principles of the reduction processes. In case of data, fuzzy clusters reveal a structure in data and quantifying the assignment through membership

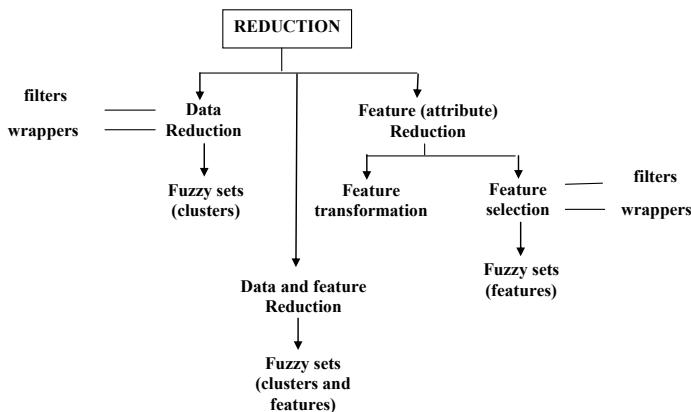


Fig. 8.12. Categories of reduction problems in pattern recognition: feature and data reduction, use of filters and wrappers criteria.

grades. The same clustering mechanism could be applied to features and form collections of features that could be treated en masse. The concept of bi-clustering engages the reduction processes realized together and leads to a collection of fuzzy sets described in the Cartesian product of data and features. In this sense, with each cluster one associates a collection of the features that describe it to the highest extent and make it different from the other clusters.

8.8. Conclusion

We have presented a motivation for the use of fuzzy sets as a logic canvas for pattern recognition problems possessing imprecise or vague information. Fuzzy set-based logic processors and Petri nets are natural extensions of their crisp binary counterparts. The former carry much stronger logic connotations than architectures such as radial basis function neural networks. The latter possess learning characteristics similar to neural networks without a reduction in the knowledge representation capabilities of the binary Petri net. Finally, context-based partially supervised extensions to the fuzzy c-means clustering algorithm were also presented. By adjusting the optimization function to take into account a fuzzy set-based context, confidence (as determined by an external source such as a domain expert) of assigned class labels may be exploited to more accurately determine the intrinsic structure of data.

8.9. Acknowledgment

Support from the Natural Science and Engineering Research Council of Canada (NSERC) is gratefully acknowledged.

References

- [1] R.E. Bellman, R. Kalaba and L.A. Zadeh, Abstraction and pattern recognition, *J. Mathematical Analysis and Applications*, 13,1–7 (1966).
- [2] A. Kandel, *Fuzzy Mathematical Techniques with Applications*, Addison-Wesley, New York, (1986).
- [3] W. Pedrycz and F. Gomide, *An Introduction to Fuzzy Sets: Analysis and Design*, MIT Press, Cambridge (1998).
- [4] R. P. W. Duin, The use of continuous variables for labeling objects, *Pattern Recognition Letters*, 1, 15–20 (1982).

- [5] R.P. Li, M. Mukaidono and I.B. Turksen, A fuzzy neural network for pattern classification and feature selection, *Fuzzy Sets and Systems*, 130, 101–108 (2002).
- [6] M. Roubens, Pattern classification problems and fuzzy sets, *Fuzzy Sets and Systems*, 1, 239–252 (1978).
- [7] T. Terano, K. Asai and M. Sugeno. *Fuzzy Systems Theory and Its Applications*, Academic Press, Boston (1992).
- [8] M.G. Thomason, Finite fuzzy automata, regular fuzzy languages, and pattern recognition, *Pattern Recognition*, 5, 383–390 (1973).
- [9] H.J. Zimmermann, *Fuzzy Set Theory and Its Applications*, Kluwer, Boston (1991).
- [10] J.C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, New York (1981).
- [11] R.O. Duda, P.E. Hart and D.G. Stork, *Pattern Classification*. 2nd Edition, John Wiley & Sons, New York, (2001).
- [12] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd Edition, Academic Press, Boston (1990).
- [13] F. Hoppen, F. Klawonn and R. Kruse, and T. Runkler *Fuzzy Cluster Analysis*, John Wiley & Sons, New York, (1999).
- [14] G.J. Klir and T.A. Folger *Fuzzy sets, Uncertainty, and Information*, Prentice Hall, Englewood Cliffs, (1988).
- [15] W. Pedrycz and F. Gomide, *Fuzzy Systems Engineering: Toward Human-Centric Computing*, Wiley-IEEE Press, Hoboken (2007).
- [16] A. Bargiela and W. Pedrycz, *Granular Computing: An Introduction*, Kluwer, Boston (2009).
- [17] W. Pedrycz, *Granular Computing: Analysis and Design of Intelligent Systems*, CRC Press, Boca Raton (2013).
- [18] L.A. Zadeh, Toward a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic, *Fuzzy Sets and Systems*, 90, 111–117 (1997).
- [19] R.E. Moore, *Interval Analysis*, Prentice-Hall, New York, (1966).
- [20] L.A. Zadeh, Fuzzy sets, *Information Control*, 8, 338–353 (1965).
- [21] L.A. Zadeh, Fuzzy logic = Computing with words, *IEEE Trans. Fuzzy Systems*, 4, 103–111 (1996).
- [22] L.A. Zadeh, Toward a generalized theory of uncertainty (GTU)—an outline, *Information Sciences*, 172, 1–40 (2005).
- [23] Z. Pawlak, Rough Sets, *Intl. J. Computer and Info. Sciences*, 11, 341–356 (1982).
- [24] Z. Pawlak, *Rough Sets: Theoretical Aspects of Reasoning About Data*, Kluwer, Dordrecht, (1991).
- [25] Z. Pawlak and A. Skowron, Rudiments of rough sets, *Information Sciences*, 3–27 (2007).
- [26] B. Ripley, *Pattern Recognition and Neural Networks*, Cambridge University Press, Cambridge (1996).
- [27] W. Pedrycz, and A. Rocha, Fuzzy-set based models of neurons and knowledge-based networks, *IEEE Trans. Fuzzy Systems*, 1, 254–266 (1993).

- [28] C. Shannon, A symbolic analysis of relay and switching circuits, *Trans. American Institute of Electrical Engineers*, 57, 713–723 (1938).
- [29] P.K. Simpson, Fuzzy min-max neural network. I. Classification, *IEEE Trans. Neural Networks*, 3, 776–786 (1992).
- [30] P.K. Simpson, Fuzzy min-max neural network–Part 2: Clustering, *IEEE Trans. Fuzzy Systems*, 1, 32–45 (1993).
- [31] S. Abe, Dynamic cluster generation for a fuzzy classifier with ellipsoidal regions, *IEEE Transaction on Systems, Man, and Cybernetics*.
- [32] A. Rizzi, F.M.F. Mascioli and G. Martinelli, Generalized min-max classifier, *Proc. Ninth Intl. Conf. Fuzzy Systems*, May 7–10, San Antonio, USA 36–41 (2000).
- [33] A. Rizzi, M. Panella and F.M.F. Mascioli, Adaptive resolution min-max classifiers, *IEEE Trans. Neural Networks*, 13, 402–414 (2002).
- [34] B. Gabrys and A. Bargiela, Generalized fuzzy min-max neural network for clustering and classification, *IEEE Trans. Neural Networks* 11, 769–783 (2000).
- [35] W. Pedrycz, and G. Succi, Genetic granular classifiers in modeling software quality, *J. Systems and Software*, 76, 277–285 (2005).
- [36] W. Pedrycz, B.J. Park and S.K. Oh, The design of granular classifiers: A study in the synergy of interval calculus and fuzzy sets in pattern recognition, *Pattern Recognition*, 41, 3720–3735 (2008).
- [37] W. Pedrycz and F. Gomide, A generalized fuzzy Petri net model, *IEEE Trans. Fuzzy Systems*, 2, 295–301 (1994).
- [38] S.K. Pal and S. Mitra, *Neuro-Fuzzy Pattern Recognition*, Wiley, New York, (1999).
- [39] W. Pedrycz, Conditional fuzzy clustering in the design of radial basis function neural networks, *IEEE Trans. Neural Networks*, 9, 601612 (1998).
- [40] N.J. Pizzi, Bleeding predisposition assessments in tonsillectomy/ adenoidectomy patients using fuzzy interquartile encoded neural networks, *Artificial Intelligence in Medicine*, 1–3, 65–90 (2001).
- [41] H. Takagi and I. Hayashi, NN-driven fuzzy reasoning, *Intl. J. Approximate Reasoning*, 5, 191–212 (1991).
- [42] T. Kohonen, *Self-Organization and Associative Memory*, Springer-Verlag, New York, (1989).
- [43] S. Grossberg, Adaptive pattern classification and universal recoding 1. Parallel development and coding of neural feature detectors, *Biol. Cyber.* 3823, 121–134 (1976).
- [44] E. Backer, Computer Assisted Reasoning in Cluster Analysis, Prentice Hall, Englewood Cliffs (1995).
- [45] A.K. Jain and R.C. Dubes *Algorithms for Clustering Data*, London: John Wiley & Sons, New York, (1988).
- [46] N.J. Pizzi, A fuzzy classifier approach to estimating software quality, *Information Science*, 241, 1–11 (2013).

- [47] L.A. Zadeh, K. Fu, K. Tanaka and M. Shimura, *Fuzzy Sets and Their Applications to Cognitive and Decision Processes*, Academic Press, London (1975).
- [48] N.J. Pizzi and W. Pedrycz, Fuzzy set theoretic adjustment to training set class labels using robust location measures, *Proc. IEEE Intl. Conf. Neural Networks*, July 24–27, Como, Italy, 109–112 (2000).
- [49] W. Pedrycz, Conditional fuzzy c-means, *Patt. Recogn.Letters* 17, 625–632 (1996).
- [50] L.A. Zadeh, Fuzzy sets and information granularity, In Gupta, M., Ragade, R. K. and Yager, R. R. (1979), *Advances in Fuzzy Set Theory and Applications*, North Holland, Amsterdam, 3–18 (1979).
- [51] M. Daszykowski, B. Walczak and D.L. Massart, Representative subset selection, *Analytica Chimica Acta*, 468, 91–103 (2002).
- [52] F. Marecelloni, Feature selection based on a modified fuzzy C-means algorithm with supervision, *Information Sciences* 151, 201–226 (2003).
- [53] O. Uncu and I.B. Turksen, A novel feature selection approach: Combining feature wrappers and filters, *Information Sciences*, 117, 449–466 (2007).

Chapter 9

Optimizing Neural Network Structures to Match Pattern Recognition Task Complexity

B. G. Gherman, K. Sirlantzis and F. Deravi

*School of Engineering and Digital Arts
University of Kent, Canterbury, UK*

F.Deravi@kent.ac.uk

Feed-forward neural networks are used to learn patterns in their training data. However, the problem of estimating the required size and structure of the network is still not solved. If we choose a network that is too small for a particular task, the network is unable to “comprehend” the intricacies of the data. On the other hand if we choose a network that is too big for the complexity of our problem, there is a danger of over-training. Therefore, we investigate possible ways to find the appropriate size for a modular feed-forward neural network for a given training set. Furthermore, we adopt a paradigm used by the Roman Empire and employed on a wide scale in computer programming, which is the “Divide-et-Impera” approach, to divide a given dataset into multiple sub-datasets, solve the problem for each of the subsets and fuse the results to form a solution for the initial complex problem as a whole. Our results show that, using this approach, for certain classes of pattern matching problems, it is possible to identify suitable structures that match the task complexity and thus optimize resource usage while maintaining performance levels.

9.1. Challenges in Designing Pattern Recognition Systems

Information is ubiquitous as the atmosphere we are breathing. The ancient Greek philosophers like Plato [1] and Aristotle [2] have realized this more than two thousand years ago, that we are perpetually surrounded by information, and what is even more interesting is, that they also formulated the fact that information should be grouped into categories. The process of learning is tightly related with grouping similar information so as to form recurring patterns and conversely, dispersing information is used to distinguish and discriminate information.

The early philosophers didn't stop at dreaming just about how to encode information but, they imagined how to construct machines with human-like abilities. These ideas were probably early the sparks that ignited the development of early calculating machines, *Automatons*, computers and which will probably lead to Generalized Artificial Intelligence.

An outstanding account of the history of Artificial Intelligence is to be found in Nils J. Nilsson's book, entitled: "The Quest for Artificial Intelligence" [3], where he shows the timeline of events that have led to the current state of Artificial Intelligence as we know it.

Information is around us, if we care to encode it either consciously or unconsciously, therefore there is a natural labelling of objects with pieces of information that requires two more processes namely, storing and retrieving these labels.

Conversely, recognizing patterns belonging to one or more categories is associated with identification and classification of newly sensed information, which is the main aim of pattern recognition.

The problem of concern for pattern recognition is to build a system which assigns newly acquired measurement examples to one of k categories or classes. This process is accomplished by first modelling or learning the possible underlying characteristics of a set of known data that has already been categorised correctly into k classes. This type of learning is called supervised learning and the process involved in learning is called training a classifier. A classifier is the realization of a learning algorithm. An intricate classifier may employ several learning algorithms once, such is the case in ensemble learning, but the relationship is of the one-to-many type.

Dietrich [4] makes a clear distinction between a classifier and a learning algorithm which is very important, in the context of evaluating the performance of classifiers which actually is being done, since the assessment of classifiers implies using a learning algorithm. This distinction between classifiers and learning algorithms is analogous to the distinction between an object and its class in Object Oriented Programming. Classifiers are manifestations or incarnations of learning algorithms. Hence, the two notions are closely related.

The overall goal of pattern recognition is to build classifiers with lowest possible error rates that work well for a wide variety of input data.

This poses a problem straight away, for how can we compare the performance of one learning algorithm with that of another learning algorithm?

The answer to this question is elusive. Since, there is no feasible way of acquiring all of the values of the population from where the data arises. We

can only extract samples from that population, which in sometimes cases will have only small number of specimens.

Nonetheless, there are methods of approximating the error a classifier makes on the whole population. To this effect, there are published works that describe the methodology for comparing the performance of two classifiers or learning algorithms one of which is Dietterich's paper [4], which states that is none of the statistical tests described in the paper, some of them widely used and evaluated in the same paper that can answer the question weather one algorithm will produce more accurate classifiers when compared against another learning algorithm. Kuncheva [5] re-iterates the same findings in her book "Combining Pattern Classifiers".

There are multiple reasons for the unreliability of the statistical tests to compare the performance difference between two classifiers described by Dietterich [4], Kuncheva [5], namely:

- all of the statistical test require using holdout or re-sampling methods, which reduce the number of available examples in the training set to be smaller than the total number of examples that are labelled and available at the time of assessment, which in turn is much smaller than all the possible input combinations of the classifier;
- the statistical test require a number of independence assumptions and some normality assumptions which are often violated;
- the statistical test that use holdout methods for dividing the existing data into training and testing sets do not measure the variation resulting from the choice of training/testing sets nor do they measure the internal randomness of the algorithm;
- the choice of the training and testing sets has a major impact;
- miss-labelled ground truth data can be present in the testing sets and this contamination introduces an error constant. The classifier cannot have an error lower than this constant.

As a consequence of these problems the performance comparisons between two classifiers have to be viewed as approximate, heuristic tests, rather than rigorously correct statistical methods, that are not universally correct, regardless of what type of data was used to evaluate the two classifiers. But, on the contrary, some classifiers are better suited than others for one particular arrangement of the input data. For example, linear or quadratic classifiers are very well suited for data that has normally distributed sample values, and more importantly the shape of the decision surface between the k -classes has a particular shape. Is an m -dimensional hyperplane line for

the linear classifier or a quadratic in m -dimensional space for the quadratic classifier, where m is the dimensionality of the data. These two examples of classifiers will be incapable of efficiently model the intricacies of input data that has a spatial organization like the dataset shown in Figure 9.1.

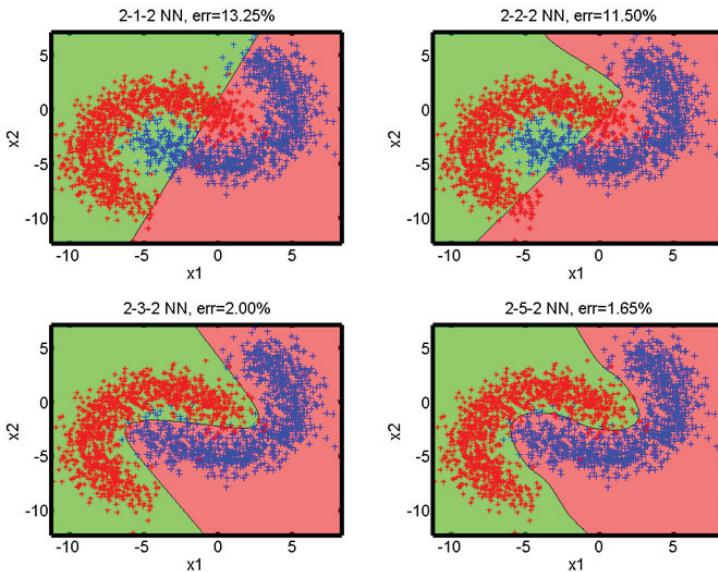


Fig. 9.1. Decision boundary produced by four NNs with 1,2,3 and 5 hidden neurons.

9.1.1. Research Problem and Motivation

The present chapter will investigate the issues related in designing a Modular Artificial Neural Network (MANN) for pattern recognition. The goal that we persuade is two fold.

Firstly, we desire to tackle the problem of selecting the architecture for a neural network, by seeking the answer to the following question:

“What is the number of hidden nodes a Neural Network architecture requires in order to classify a given dataset (in some sense) optimally?”

Secondly, we address the problem of module selection in a greater scenario of building a modular classifier based individual Neural Networks, whose architecture was determined by the process that we stated as our first goal.

Let us elaborate our first goal a bit more in the following paragraphs.

The question of “*how many hidden nodes are required in a Neural Network to classify a given dataset optimally?*”, is not easy to answer. Several issues arise straight away from this single question: What are the hidden nodes? What is the architecture of Neural Network? but more importantly: What is optimal classification? We shall try to provide an answer to these questions.

This chapter is structured in the following way, after this current introductory section we are going to present in Section 9.2 the background to Neural Networks. Next we will present the Data Complexity Meta-Measurements devised and calculated to describe the complexity required to classify a given dataset and the datasets that were used in experiments throughout the rest of the chapter. The third section will show how to select initial parameters for Neural Network training based on approximating the rough decision boundary. The fourth section will present the experimental setup and results achieved for Modular Neural Networks on synthetic and realistic datasets. Finally, in Section 9.5 we shall present the final conclusions resulting from our research and a few future development suggestions.

9.2. Artificial Neural Networks for Pattern Classification Applications

Neural Networks (NNs) are part of the machine learning scientific discipline, which is concerned with the design and development of algorithms that allow computers to learn based on data, such as from sensor data or off-line databases. They are suitable for performing the following types of general tasks: pattern classification, regression, and clustering [6].

There are several paradigms of learning which apply to pattern recognition [6], [7]:

- supervised learning;
- unsupervised learning;
- reinforcement learning and active learning.

We are going to employ in our research the first paradigm of learning, namely the supervised learning paradigm.

The goal of NNs is not to learn an exact representation of the training data itself, but rather build a model of the process that generates the data. This problems is said to be well posed if an input always generates an unique output and the mapping is continuous.

Supervised learning is an ill posed problem, given the training or input examples that are a subset of the Input-Output relationship, an approximate mapping is needed to be estimated. However, the input data might be noisy, imprecise and it might be insufficient to uniquely construct the mapping.

Regularization techniques can transform an ill-posed problem into a well posed one, in order to stabilize the solution by adding some auxiliary, non-negative functional of constraints [8], [9].

In this research we are concerned with solving pattern classification tasks using Neural Networks (NNs). The NNs are adapting their internal parameters (network connection weights) to model their output according to the input from the environment they are taking the information from. This process is generally called “learning” of the NN.

The parameters can be learned in three different paradigms:

- Using prescribed outputs for some of the inputs (desired outputs) which are evaluated as correct or incorrect by a third party, this is called Supervised Learning, and the input/output data is said to be labelled;
- Using a task independent measure of quality that will require the network to behave in a self organizing manner, which is called Unsupervised Learning;
- The Input to Output mapping of the network is performed continuously in order to improve on scalar performance measure, this is called Reinforcement Learning. The data in the later two paradigms is said to be unlabeled and in contrast with the labelled data, this kind of data is readily available and inexpensive to obtain.

The manner in which the neurons are organized is closely related with the learning algorithm used to train the network. According to Mehrotra *et al.* [10], Neural Networks can be categorized according to their topology, i.e. the way neurons are organized as the following types:

- Fully connected Networks
- Single Layer Networks feed-forward Networks
- Multilayer feed-forward Networks
- Acyclic Networks
- Modular Neural Networks

We will need to add that feedback links can exist between the nodes, so that the outputs of a particular node are connected to nodes from which the particular node receives a connection. According to Haykin [6] these types

of networks are called Recurrent Networks. In the next two sections we will deal with two types of NNs, namely Feed-forward and modular networks, which are the main types of networks in our research.

9.2.1. Feedforward Backpropagation Networks

Backpropagation, or propagation of error, is a common method of teaching artificial neural networks how to perform a given task. It was first described by Paul Werbos in 1974, but it wasn't until 1986, through the work of David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams, that it gained recognition, and it led to revival of the research of artificial neural networks.

According to Simon Haykin [6], the backpropagation algorithm has emerged as the workhorse for the design of feedforward networks known as multilayer perceptrons (MLP).

As shown in Figure 9.2, a multilayer perceptron has an input layer of source nodes and an output layer of neurons (i.e., computation nodes); these two layers connect the network to the outside world. In addition to these two layers, the multilayer perceptron usually has one or more layers of hidden neurons, which are so called because these neurons are not directly accessible. The hidden neurons extract important features contained in the input data.

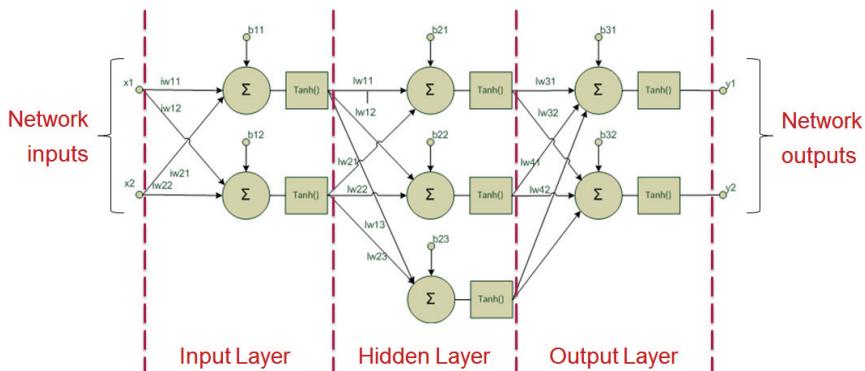


Fig. 9.2. Example of a multi layered, fully connected, Feed-Forward Neural Network architecture with 3 layers, 2 inputs, 3 hidden nodes and 2 outputs, generally called a 2-3-2 network.

The backpropagation learning algorithm is simple to implement and somewhat computationally efficient in that its complexity is linear in the connection weights of the network. However, a major limitation of the algorithm is that it does not always converge or it can be extremely slow, particularly when we have to deal with a difficult learning task that requires the use of a large network.

Despite these shortcomings of the MLP architecture and training algorithms, these networks are used in solving problems where the dimensionality is small and the amount of training data is sufficient so that the “Curse of dimensionality” is avoided, as for example in the recent work published by: Foody [11] in 1995, Blamire [12] in 1996 and more recently in 2003 the work of Pal and Mather [13]. The performance of the MLP is dependent on the quality of the training data, a fact that was neglected a bit by previous studies but Taskin Kavzoglu [14] in 2009 has improved the classification accuracy by 2-3 percent by eliminating outliers, using training set refinement, supports the premise that MLPs are still viable to solve current problems.

For a good generalization the number of examples in the training set N , has to be several times larger than the neural network’s capacity [15]:

$$N \gg \frac{N_w}{N_y}$$

where N_w is the total number of weights or free parameters and N_y is the total number of output units.

9.2.2. Modular Neural Networks

The best description of the Modular Neural Networks (MNN) has been given by Ronco and Gawthrop in their technical report from 1995 [16], where they define the MNN in comparison with “global” backpropagation and clustering neural networks. Here they state that a modular network is the most likely to combine the desirable features of the two aforementioned classes of networks. It is pointed out that a MNN has to have its modules assigned problem specific sub tasks, and not just any sub-task, resulting from an arbitrary decomposition scheme that might not have any physical meaning. The reason for dividing the problem, that we want to solve, into sub tasks that are physically meaningful, is desirable from at least two stand points:

- the number of variable parameters is reduced to a number large enough to provide a good solution;

- the network becomes tractable, so that the internal workings of the network have some meaning attached and are not just black boxes that cannot be inspected easily.

Further in this report and in the survey of Auda and Kamel [17], we find the steps needed to be accomplished by the designer of a MNN:

- Decomposition of the main task into sub-tasks;
- Organization of the modular architecture;
- Communication between the modules and decision fusion.

Modular neural networks possess the conceptual means to overcome some of the shortcomings of monolithic backpropagation multi-layered networks, (such as non-convergence, spatial and temporal crosstalk [6]) and use the benefits of clustering networks.

Previous research on MNNs is assessed in a very good and concise manner in the referenced papers [16], [17] and the references therein. Some problems still remain to be solved so the modular networks can be efficiently implemented for various applications, namely:

- How to split the input space properly, such that the decomposition can be beneficial to both learning and generalization;
- How to decide the proper number of experts in a committee machine for a particular task.

However, the problem of task decomposition was studied in the paper produced by Lu and Ito [18], where they consider that task decomposition can be roughly divided into three classes as follows:

- Explicit Decomposition, before learning the problem is divided into a set of sub-problems by the designer of the system using domain and a priori knowledge about the problem. The difficulty lies with the fact that a large amount of prior knowledge is required and also this knowledge has to be properly translated to the system;
- Class Decomposition, before learning the problem is broken down into a set of sub-problems so that a problem having K classes is divided into K, two class problems; An example is presented in reference [18] where a significant gain in performance is achieved compared with classical global neural networks;
- Automatic Decomposition, where a problem is decomposed into a set of sub-problems by the process of learning. The former two methods are more efficient because the task is decomposed before the learning,

but this later method is more general since it does not require any prior knowledge. Most of the automatic methods fall into this category, for instance: the mixture of experts [19] and the multi-sieving network.

The multi classifier system presented here [19] describes the usage of a MNN to recognize shapes in an industrial robotic vision project. Another example of the classification discriminatory power of MNNs is presented in [20] where the results suggest that MNNs achieved comparable to Support Vector Machines while the proposed method was slightly surpassed by Exponential Radial Basis Function kernels, the method did prove to be better than the SVM using Gaussian radial basis function.

9.3. Data Complexity

Complexity is an illusive word, we seem to understand straight away what is meant when the word arises in a conversation, yet there is no useful quantitative definition of the word.

Following the work of Tin Kam Ho and Mitra Basu in 2002 [21] and their later book of Ho, Basu and Law [22] we are describing the complexity of the datasets based on measurements calculated on the training data itself.

We set out to investigate methods of finding a suitable classifier for the training data that is available to estimate its parameters, this was hinted by previous published work of Cano in 2013 [23], Cavalcanti 2012 [24] and Sotoca in 2006 [25].

However, we did not find a reference to an automated system that would select a suitable classifier based on measurements obtained from the training data. These measurements will be referred to in this document as meta-measurements, because they are introducing another abstraction layer between the data and the actual classifier that is supposed to do the classification task. This layer of abstraction is supposed to evaluate the input training data and select a suitable classifier or in the case of modular neural networks make suggestions to alter the architecture of such modules in a neural network.

Therefore, in Table 9.1 we list the meta-measurements obtained from the data, which can be seen as complexity measurements, since they will be able to distinguish, for example in the simplest case, between a linearly separable dataset and a dataset which has a decision boundary of a second order. The later dataset will obviously have a higher complexity than the former. Details of these metrics can be found in [26].

Table 9.1. List of Meta-Measurements.

Nr.	Name	Description
1	F1(*)	Multi-dimensional Fisher's discriminant ratio
2	F2(*)	Volume of overlap
3	F3(*)	Feature efficiency
4	CL1(*)	LD classifier error rate
5	CQ1(*)	QD classifier error rate
6	LL1(*)	LD classifier non-linearity
7	LQ1(*)	QD classifier non-linearity
8	LK1(*)	KNN classifier non-linearity
9	VC	Vector Correlation between features and labels
10	E1	Average Shannon entropy
11	E2	Average MAXIMUM variance entropy
12	E3	Total Shannon entropy
13	E4	Total MAXIMUM variance entropy
14	FE1	Feature evaluation criterion, Inter-Intra distance
15	FE2	Sum of Mahalanobis distances
16	FE3	Minimum Mahalanobis distances
17	FE4	Sum of squared Euclidean distances
18	FE5	Minimum of squared Euclidean distances
19	FE6	1-Nearest Neighbour Leave-One-Out performance
20	S1	Average Skewness
21	S2	Maximum Skewness
22	S3	Minimum Skewness
23	S4	Minimum absolute Skewness
24	S5	Average Kurtosis
25	S6	Maximum Kurtosis
26	S7	Average inter-feature correlation
27	S8	Maximum Inter-feature correlation
28	S9	Minimum Inter-feature correlation
29	S10	Average feature to label correlation
30	S11	Minimum absolute feature to label correlation
31	S12	Maximum absolute feature to label correlation
32	S13	STATLOG γ
33	S14	STATLOG M
34	S15	STATLOG SD_ratio
35	G1	Boundary rotation angle
36	G2	Number crossings of the decision boundary with itself
37	G3	Length of the extracted decision boundary
38 - 53	A1 - A16	Decision boundary angle profile

In this table the meta-measurements which have a star in parenthesis (*) besides their shorthand name have been inspired from Ho and Basu's paper from 2002 [21] with slight modifications [26]. The other meta-measurements are grouped together by the first one or two letters in their name. These have a capital "E" to denote entropy measurements, the capital letters "FE"

for feature evaluation measurements, “S” for statistical measurements, “G” for geometrical measurements and finally “A” for the angle profile measurements. The vector correlation measurement stands alone in its own group named “VC”.

An important note about all of employed meta-measurements is that, they all work with datasets having at most two classes, they are not adapted to work with more than two classes. However, by splitting the k -class problem either in a “1 versus ALL” or “1 versus 1” fashion they are adapted to work even for this type of datasets.

9.3.1. *Synthetically Generated Data*

In order to characterize the spatial distribution of the data-points in a pattern recognition dataset arriving to a pattern recognition system we have decided to use polynomials that can be superimposed to the decision boundary of the input data. Hence, in order to assess the performance of the systems we have developed, we needed some “ground truth” data that had polynomial decision boundaries of a known polynomial order. Therefore, we generated synthetically such datasets with the orders of the polynomials ranging from 1 to 10. These datasets will be discussed in the remainder of this section.

Before we dwell into the description of the algorithms used to generate the synthetic datasets, we have to stop to define what decision boundaries are.

A decision boundary is characteristic for a given dataset and a given classifier. Any classifier produces a decision boundary in the feature space of the given problem.

A decision boundary of a given pattern recognition problem is the locus of points in the feature space of the pattern recognition problem that has equal posteriori probability produced by a given classifier. Therefore, if a new sample that is to be classified lies exactly on the decision boundary the classifier will not have enough information to assign it to one class or the other. It will assign a class label in a deterministic or random fashion, not based on any factual or learned information. However, this case is rather infrequent and does not present a major impediment in the pattern recognition world. The reason why we are concerned with decision boundaries is because any classifier produces a decision boundary.

We have generated datasets of two classes and two features (i.e. a 2-dimensional dataset) where the decision boundary formed between the two

classes is a polynomial of a known order $n = [1, \dots, 10]$.

The employed polynomials have the following general form:

$$P(x) = \sum_{i=0}^n a_i x^i = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x^1 + a_0 x^0$$

where a_i are the coefficients of the polynomial and n is the highest power of x which appears in $P(x)$ or the order of the polynomial.

The data that we used in our experiments was artificially generated in order to have only two classes and a separating decision boundary between the two classes, a polynomial curve of a known order ranging from 1 to 10. Also, the data points that are generated have only 2 features (or a dimensionality of 2), therefore it can be represented in 2 dimensions and graphed easily.

The datasets that were generated can be grouped in two categories based on the distribution of the data points, as follows:

- Uniformly distributed random feature values;
- Normally distributed random feature values, that form Gaussian clouds around points lying on each side of the polynomial decision boundary.

Within each of these groups we have generated datasets with: positive separation (i.e.: more pronounced separation), negative separation (i.e.: less separation and more pronounced overlap in some cases) or neutral separation distance from the decision boundary itself.

We have generated 6,000 datasets, which consist of datasets having polynomial orders from 1 to 10 and 100 datasets with the same order, in two categories of data point distributions (Uniform and Gaussian clouds) with 3 degrees of separation.

The flowchart of the algorithm used to generate all the synthetic datasets can be seen in Figure 9.3. In this figure the blocks coloured in light blue represent algorithmic components, respectively the blocks coloured in green represent data outputs. From this figure we can see that the algorithm has 3 common algorithmic components that generate and validate the polynomial $P(x)$, these are numbered 1, 2 and 3, with the generated polynomial is numbered 4 in Figure 9.3. Whereas, for each category of datasets (i.e. Uniform and Gaussian Clouds) the final 2 algorithmic blocks are different. That is to say that in order to generate a dataset of the Gaussian Cloud type (labelled 7a) we need to follow steps 5a and 6a respectively to generate a Uniformly distributed dataset (labelled 7b) we need to follow steps 5b and 6b.

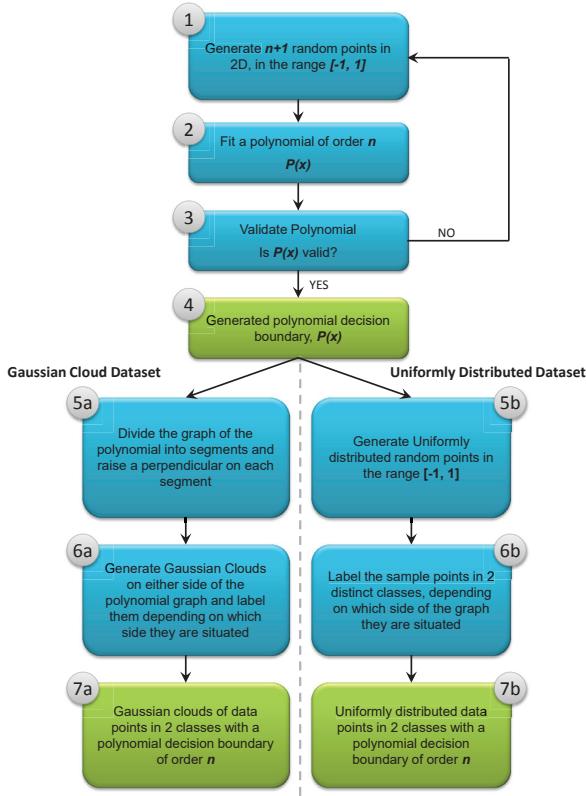


Fig. 9.3. Flow chart of the synthetic datasets generation algorithm.

In the following paragraphs we shall discuss the details of the common pathway of generating the synthetic datasets, while the details pertaining each dataset type will be discussed in Sections 9.3.1.2 and 9.3.1.1.

The first step of the algorithm to generate the synthetic datasets depicted in Figure 9.3, is to obtain $n + 1$ random pairs of coordinates in the 2-dimensional plane (x_k, y_k) where $x_k, y_k \in [-1, 1]$ and $k = 1, \dots, n + 1$.

These points are used in the second step which interpolates upon these $n + 1$ points a unique polynomial $P(x)$ of order n .

The third step in the algorithm validates the polynomial by performing the following checks:

- Sample points along the graph of the polynomial $P(x)$, and make sure there are enough points that fall in the region bounded by -1 and 1 on either dimensions;

- The graph of the polynomial $P(x)$ divides the region of the plane bounded by -1 and 1 on either dimensions into roughly the equal regions, without a major imbalance;
- Verify that the number of tangent lines to graph of the polynomial is not less than n .

If all of these checks are passed, the polynomial $P(x)$ is deemed to be suitable for later use as a decision boundary for the synthetic dataset that is to be generated in both of the dataset categories mentioned above. When any of these checks fails, the polynomial will be abandoned, and a new one will be generated by jumping to the first step in the flowchart shown in Figure 9.3 and the process is repeated until a polynomial is found to pass all these checks. Rejecting a polynomial happens rather infrequently, but these checks are used to filter out malformed or trivial polynomials.

The tangent lines to the graph of the polynomial that are used in the validation checks mentioned above are exemplified in Figure 9.4. Since, there are infinite number of tangent lines to the graph of the polynomial, we have chosen to consider only a representative number of tangents to the graph, that is equal to n the order of the polynomial.

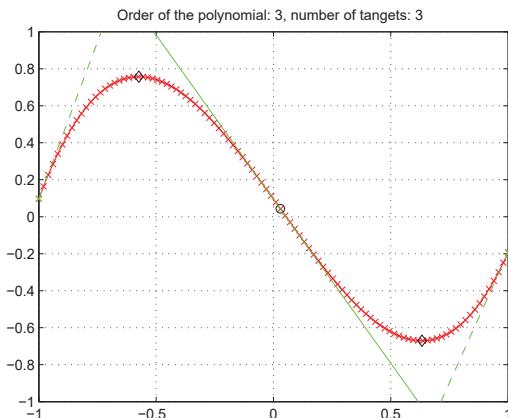


Fig. 9.4. Plot of a third order polynomial, with its characteristic tangent lines.

The considered tangent lines are the following:

- 2 tangent lines at the border of the graph, i.e. at the points $(-1, P(-1))$ and $(1, P(1))$. These are shown in Figure 9.4 as green dashed lines;
- Respectively, the tangent lines at the point of inflection of the graph of

the polynomial. That is, at the location $(x_{\text{inflection}}, P(x_{\text{inflection}}))$ where the second order derivative of $P(x)$ vanishes. This tangent line is shown in Figure 9.4 as the green full line.

The specifics generation of the datasets that have a polynomial boundary of a given order n is discussed next, in the following section.

9.3.1.1. Gaussian Cloud Datasets

We have generated Gaussian Cloud datasets with 3 types of separation between the two classes, namely:

- Neutral separation;
- Increased or positive separation;
- Decreased or negative separation.

In order to describe the generation algorithm for the datasets of a Gaussian Cloud type, we have to refer to Figure 9.6.

From this figure we can observe how the datasets are generated.

Firstly, the graph of the polynomial $P(x)$ is sampled in the range $[-1, 1]$. Any points that are outside the range $[-1, 1]$ are removed.

Then, line segments are considered by taking two consecutive points from the sampled graph of the polynomial. The endpoints of the line segments are depicted in Figure 9.6 by the blue diamond symbols.

For each of these line segments, perpendiculars are raised from the midpoint of each of the line segments on either side of the line segments. The length of the perpendiculars are given by Formula 9.1 and are show in the bar plot in Figure 9.5 b), c) and d).

The endpoints of these perpendiculars form the centres of the cloud of points that will be generated. These endpoints are visible in Figure 9.6 as blue crosses with a circle around the cross.

These clouds of points will follow a Gaussian distribution:

$$X_{\text{cloud},i} \sim \mathcal{N}(\mu_i, \Sigma_i)$$

where μ_i is the coordinate of the end of the perpendicular raised on the line segment obtained from sampling the graph of the polynomial and sigma is the circular covariance matrix:

$$\Sigma_i = \begin{bmatrix} \sigma_i^2 & 0 \\ 0 & \sigma_i^2 \end{bmatrix}$$

The variance σ_i^2 is changing for each order $i = 1 \dots 10$ of the polynomial according to this formula:

$$\sigma_i^2 = 1/(200 \cdot i)$$

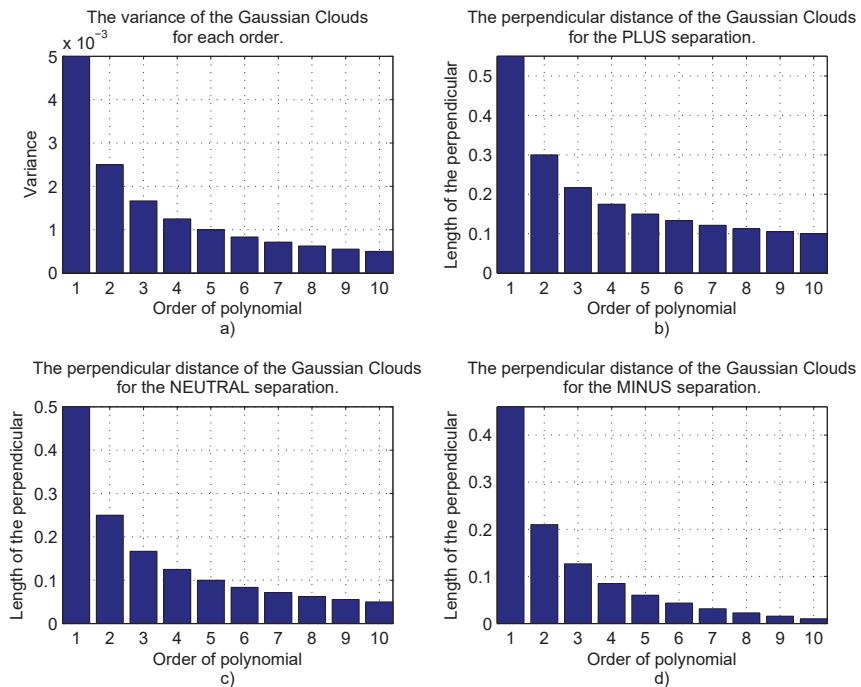


Fig. 9.5. The parameters to generate the Gaussian Cloud datasets. a) the variance σ^2 for each order; The perpendicular distance for each order b) for the increased (PLUS) separation; c) for the NEUTRAL separation; d) for the decreased (MINUS) separation.

The perpendicular distance (PD) is varying with each order of the polynomial (i) according to following formula:

$$PD = \frac{0.5}{i} + S \quad (9.1)$$

Where the separation S is equal to 0.05, 0, -0.04, in turn, depending on what kind of dataset is to be generated.

The clouds of points that are generated on one side of the line segment will be assigned one class and the points generated on the other side of the line segment will be assigned another class, which is shown in Figure 9.6 as green and red stars symbols.

This process is repeated for all the line segments the particular polynomial $P(x)$ has.

An example of Gaussian Cloud datasets are shown in Figure 9.6, where we can observe a close-up of the decision boundary for a polynomial of order 2.

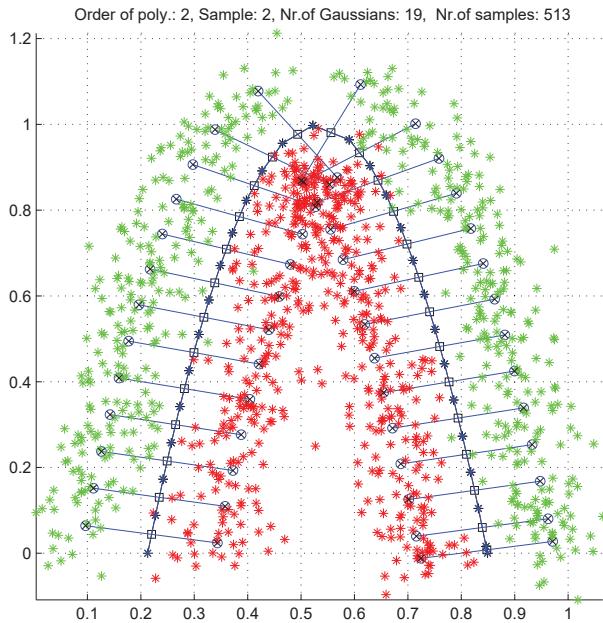


Fig. 9.6. Detail showing how the Gaussian clouds are generated, for a polynomial of order 2.

9.3.1.2. Uniformly Distributed Datasets

The generation algorithm for this dataset is outlined in Figure 9.3, and the steps labelled 1, 2, 3, 4, 5b and 6b, therefore steps 1 to 4 are already described in 9.3.1. The reminder of steps (5b and 6b) will be described in this section.

Firstly, the two dimensional plane region bounded by the range [-1, 1] is filled with points.

In the second step, the points generated in the above step are thresholded, that is, they are assigned one of two class labels depending whether they lie above or below the graph of the polynomial.

The Uniformly distributed datasets are generated with 3 types of separation:

- Neutral separation;
- Increased or positive separation;
- Decreased or negative separation.

In order to produce the datasets with increased and decreased separation the data-points of one class are shifted along the vertical dimension with a constant.

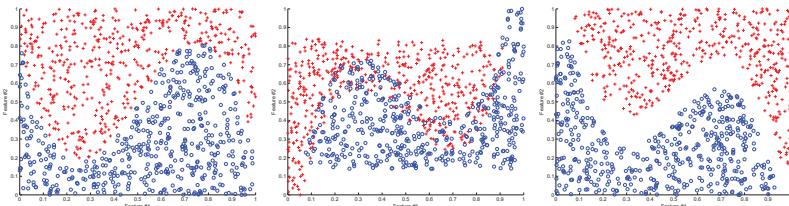


Fig. 9.7. Scatter plots of uniformly distributed set of points distributed along a polynomial boundary of order 3 and 3 degrees of separation.

9.4. Neural Network Weight Adaptation

In this section we shall talk about a technique developed by the authors, to improve the classification accuracy and reduce the number of training epochs of the feed-forward backpropagation neural networks when trained on data that has a decision boundary that is polynomial or it can be approximated to a polynomial. This has been achieved by tuning the weights of the neurons so that they reproduce the location of tangent lines to the polynomial graph at the inflection points of the graph where the second order derivative becomes zero and also the tangents to the graphs of the polynomial at the boundaries of the considered graphing ranges (e.g. [-1, 1]). These tangent lines will be referred to later in this section as “*characteristic tangent lines*”.

First we begin by describing the data we used in this endeavour which has a known polynomial order, then, we briefly highlight the way we obtained the characteristic tangent lines of the known polynomial. We use these characteristic tangent lines to tune the weights of the neural network so that the decision boundary produced by the neural network will become very close to the true boundary present in the data. Finally, we present the results that we obtained by this setup by analysing the *Total Variance Distance* of the Gaussian distribution of the classification error produced by our experiment, compared with a set of control experiment, which does not employ the weight adaptation technique of tuning the weights of the neural networks.

9.4.1. Weight Adaptation

The architecture of a neural network is defined by:

- The number of hidden nodes
- The number of layers
- The presence/absence of feedback loops
- The “connectedness” of the network (fully connected or not)
- The topological/hierarchical organisation
- The type of activation function of the nodes

We are concerned with just the first item in the list of parameters that define the architecture of the neural network, namely the number of hidden neurons in the hidden middle layer of the neural network. While we assume constant values to all the other parameters of the network architecture. The neural networks we are investigating has a single hidden layer, does not have feedback loops (it is a feed-forward network), it is fully connected, the topological layout is equivalent to a directed graph and the non-linear activation function is the hyperbolic tangent.

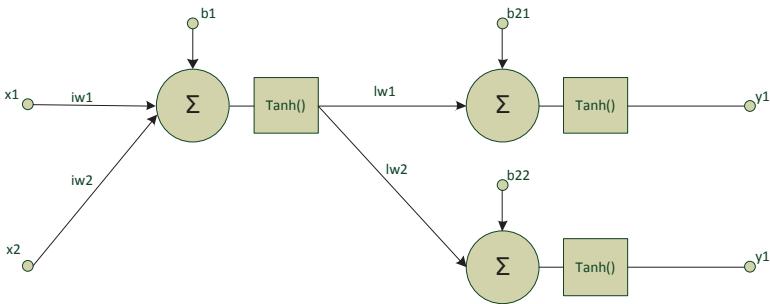


Fig. 9.8. Schematic of a 2-1-2 neural network.

$$\begin{cases} y_1 = \tanh(b_{21} + lw_1 \tanh(b_1 + iw_1 x_1 + iw_2 x_2)) \\ y_2 = \tanh(b_{22} + lw_2 \tanh(b_1 + iw_1 x_1 + iw_2 x_2)) \end{cases} \quad (9.2)$$

Solve for x_2 with respect to x_1 when $y_1 = y_2$.

$$x_2(x_1) = \frac{\frac{1}{2} \log \left(\frac{-b_{21}-b_{22}+lw_1-lw_2}{b_{21}-b_{22}-lw_1+lw_2} \right) - b_1}{iw_2} - \frac{iw_1}{iw_2} x_1 \quad (9.3)$$

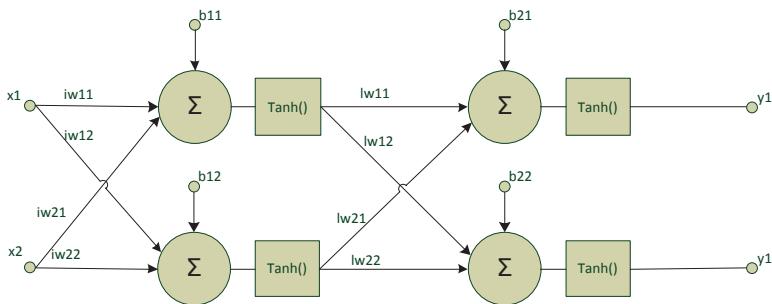
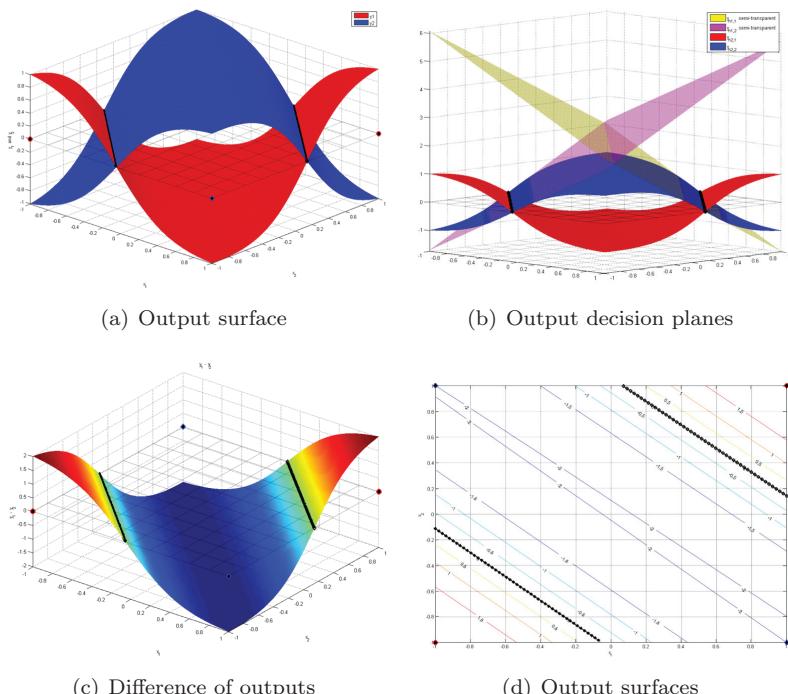


Fig. 9.9. Schematic of a 2-2-2 neural network.

Fig. 9.10. Output produced by a 2-2-2 NN trained on the XOR problem. a) surfaces produced by each individual output neuron; b) individual surfaces with decision planes produced by each neuron; c) decision surface obtained by $y_1 - y_2$; d) contour plot of the output surface $y_1 - y_2$.

$$\begin{cases} y_1 = \tanh(b_{21} + lw_{11} \tanh(b_{11} + iw_{11}x_1 + iw_{21}x_2) + \\ \quad + lw_{21} \tanh(b_{12} + iw_{12}x_1 + iw_{22}x_2)) \\ y_2 = \tanh(b_{22} + lw_{12} \tanh(b_{11} + iw_{11}x_1 + iw_{21}x_2) + \\ \quad + lw_{22} \tanh(b_{12} + iw_{12}x_1 + iw_{22}x_2)) \end{cases} \quad (9.4)$$

Solve for x_2 with respect to x_1 when $y_1 = y_2$.

This is impossible to express using algebraic operators. However if we replace the non-linear function, in our case the hyperbolic tangent, with a linear function and obtain the following system of equations:

We looked at the linear expression of the inputs to the hidden nodes:

$$\begin{cases} y_1 = (b_{11} + iw_{11}x_1 + iw_{21}x_2) \\ y_2 = (b_{12} + iw_{12}x_1 + iw_{22}x_2) \end{cases} \quad (9.5)$$

We obtain the following solutions:

$$\begin{cases} x_{2,1}(x_1) = -\frac{b_{11}}{iw_{21}} - \frac{iw_{11}}{iw_{21}}x_1 \\ x_{2,2}(x_1) = -\frac{b_{12}}{iw_{22}} - \frac{iw_{12}}{iw_{22}}x_1 \end{cases} \quad (9.6)$$

They define planes in the feature space and the intersection of these planes with the “zero plane” is approximating the decision boundary.

9.4.2. Experimental Results

We have conducted two sets of experiments to determine the usefulness of our proposed method, one experiment is the control experiment where weight adaptation is not used, the other experiment is using the weight adaptation method that we have just discussed. Both sets of experiments consist of creating 10,000 datasets of polynomial order, where each dataset is classified by a feed-forward neural network. The classification error of each individual neural network is retained then the mean and standard deviation is computed. The mean and standard deviation is then compared with the mean and standard deviation of a “control” experiment in which the weight adaptation algorithm is disabled, therefore establishing a baseline for comparison.

Before we dive into the presentation of our experimental results, we are going to make a slight detour, and present some aspects of our methodology for comparison. For this reason we are going to describe the use of the statistical total variance distance between two probability distributions.

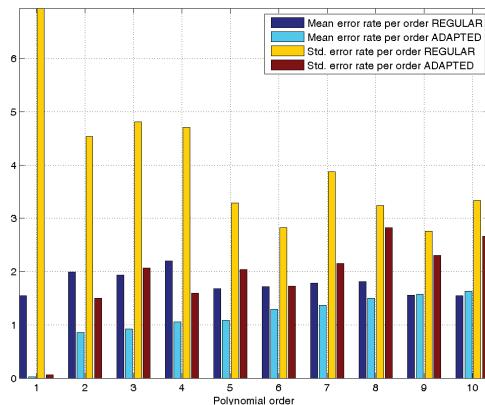


Fig. 9.11. Graphical representation of the Total Variance Distance between two Gaussian distributions.

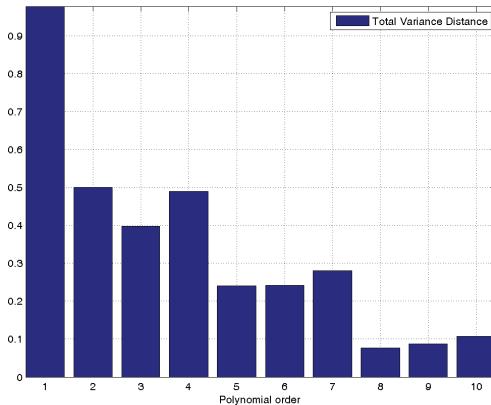


Fig. 9.12. Total Variance Distance between the probability distributions of the error rates produced with and without weight adaptation.

Lets consider that we have two Gaussian distributions of the error values of our two sets of experiments, we shall name these $e_1 \sim \mathcal{N}(0, 4^2)$ and $e_1 \sim \mathcal{N}(5, 1^2)$.

The two Gaussian probability distributions of the errors are compared using the Total Variance Distance (Equations 9.7 and 9.8), adapted from

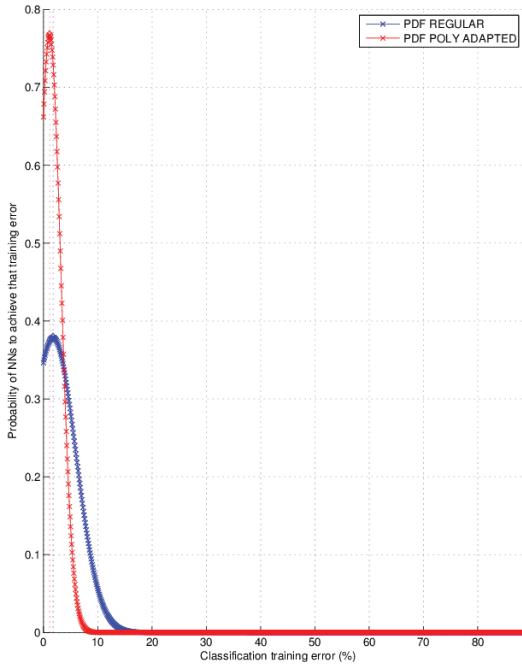


Fig. 9.13. Comparison of fitted probability distributions to the error rates produced by training 10,000 NNs without weight adaptation(blue) and with weight adapted NNs (red).

Levin, Peres and Wilmer [27]. This distance metric was chosen because it conveys the dissimilarity information between the two distributions in one single real number that belongs to the semi-closed interval $[0, 1)$, where a distance of 0 corresponds to two distributions which have exactly the same means and standard deviations, while the distance value will tend to reach the value 1 asymptotically, for the ever dissimilar Gaussian distributions, but will not reach the value 1, it will get infinitesimally close to 1. The two distributions however skewed they may be, they still have an intersection point and a very small (sometimes negligible) common overlap area.

The Total Variance Distance (or TV distance) effectively measures the common area under the two probability distribution functions.

The Total Variance distance for discrete events $\|\cdots\|_{TV \ discrete}$ is defined as:

$$\|e_1 - e_2\|_{TV \ discrete} = \frac{1}{2} \sum_{x \in \Omega} |e_1(x) - e_2(x)| \quad (9.7)$$

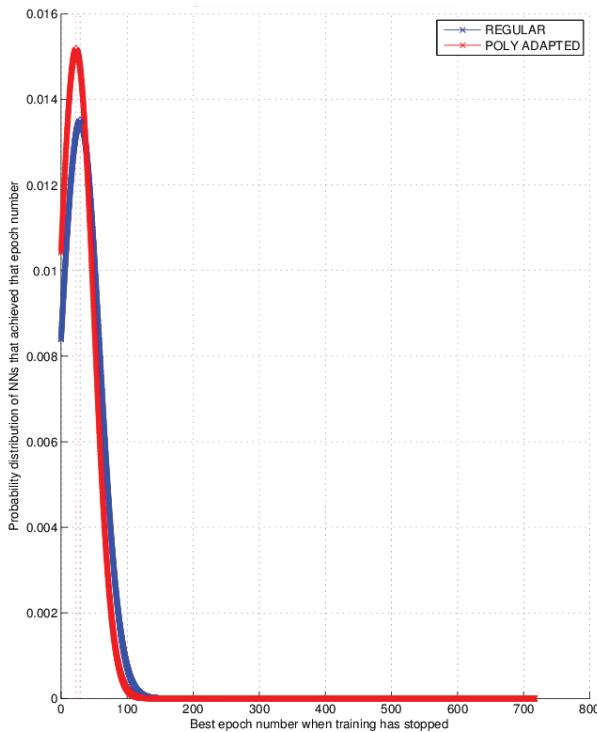


Fig. 9.14. Comparison of fitted probability distributions to the epoch reached before training was stopped for 10,000 NNs without weight adaptation(blue) and with weight adapted NNs (red).

However, we are not concerned with the measurement of the distance between discrete probability distributions, we would like to have the same measurement of distance for continuous probability distributions. Therefore, we modify the above definition by summing the absolute values between the integrals of the probability distributions on the intervals where the probability distributions have a constant relationship to one another. That is to say, that we split the interval $(-\infty, \infty)$ where we do the integration, to intervals where one particular probability distribution function is always larger than the other. The splits are therefore given by the points of intersection between the two probability distribution functions.

In order to write down the analytical form of the total variance distance for the continuous case of the probability density function we need to define the set of points of intersection between the two functions, which we will

Table 9.2. Statistics of the error distributions.

	CONTROL	ADAPTED WEIGHTS
Average value	1.7775	1.1301
Standard deviation	4.2129	2.0796
Minimum	0	0
Lower whisker	0	0
Lower 25% percentile	0.1	0.1
Median	0.5	0.5
Upper 75% percentile	1.4	1.4
Upper whisker	3.3	3.3
Maximum	88.6	65.1
Percentage in Q1	17.12%	16.95%
Percentage in Q2	57.12%	57.56%
Percentage in Q3	12.35%	17.46%
Percentage of outliers	13.41%	8.03%
Number of outliers	1,294	766
Total number of samples	10,000	10,000

Table 9.3. Statistics of the best epoch distributions.

	CONTROL	ADAPTED WEIGHTS
Average value	28.8212	22.776
Standard deviation	29.5947	26.3375
Minimum	0	0
Lower whisker	0	0
Lower 25% percentile	13	11
Median	21	16
Upper 75% percentile	34.5	26
Upper whisker	66	48
Maximum	608	717
Percentage in Q1	22.10%	21.77%
Percentage in Q2	52.90%	52.81%
Percentage in Q3	17.75%	17.89%
Percentage of outliers	7.25%	7.53%
Number of outliers	710	727
Total number of samples	10,000	10,000

later see that it will consist of mostly two points, that we shall refer to as x_1 and x_2 . Also, we will assume that $x_1 \leq x_2$.

Now we can write down the analytical form the total variance distance for continuous probability distributions $\|\cdot\cdot\cdot\|_{TV}$, as:

$$\begin{aligned} \|e_1 - e_2\|_{TV} &= \frac{1}{2} \left(\left| \int_{-\infty}^{x_1} pdf_1(x) dx - \int_{-\infty}^{x_1} pdf_2(x) dx \right| + \right. \\ &\quad + \left| \int_{x_1}^{x_2} pdf_1(x) dx - \int_{x_1}^{x_2} pdf_2(x) dx \right| + \\ &\quad \left. + \left| \int_{x_2}^{+\infty} pdf_1(x) dx - \int_{x_2}^{+\infty} pdf_2(x) dx \right| \right) \end{aligned} \quad (9.8)$$

In order to obtain the values for x_1 and x_2 , we equate the two Gaussian PDFs and solve for x to obtain the coordinates of the intersection points. Also, we observe that, generally, the two graphs have at most two intersection points since the equation is essentially a quadratic equation in x (after we apply the logarithm with base e to both sides of the equation), unless of course, the two distributions have the same σ and μ in which case the two distributions will have an infinite number of common points. However, this case is trivial and will be dealt with separately. Finally to obtain this quadratic form in x :

$$\begin{aligned} (\sigma_1^2 - \sigma_2^2) \cdot x^2 + 2(\mu_1 \sigma_2^2 - \mu_2 \sigma_1^2) \cdot x + \\ + \mu_2^2 \sigma_1^2 - \mu_1^2 \sigma_2^2 - 2\sigma_1^2 \sigma_2^2 \cdot \ln \left(\frac{\sigma_1}{\sigma_2} \right) = 0 \end{aligned} \quad (9.9)$$

By using term identification we can discover the coefficients of the quadratic equation. We will use the following notation for the coefficients of x :

$$\begin{cases} \alpha = \sigma_1^2 - \sigma_2^2 \\ \beta = 2(\mu_1 \sigma_2^2 - \mu_2 \sigma_1^2) \\ \gamma = \mu_2^2 \sigma_1^2 - \mu_1^2 \sigma_2^2 - 2\sigma_1^2 \sigma_2^2 \cdot \ln \left(\frac{\sigma_1}{\sigma_2} \right) \end{cases} \quad (9.10)$$

Hence, we obtain a standard quadratic form. Therefore, the solutions of the quadratic equation, given the notation we employed, is the following conditional set with four branches:

$$x \in \begin{cases} \left\{ -\frac{(\beta - \sqrt{\beta^2 - 4\alpha\gamma})}{2\alpha}, -\frac{(\beta + \sqrt{\beta^2 - 4\alpha\gamma})}{2\alpha} \right\} & \text{if } \alpha \neq 0 \\ \left\{ -\frac{\gamma}{\beta} \right\} & \text{if } \alpha = 0 \wedge \beta \neq 0 \\ \mathbb{C} & \text{if } \alpha = 0 \wedge \beta = 0 \wedge \gamma = 0 \\ \emptyset & \text{if } \alpha = 0 \wedge \beta = 0 \wedge \gamma \neq 0 \end{cases} \quad (9.11)$$

The first branch in (9.11) is satisfied when $|\sigma_1| = |\sigma_2|$, since the standard deviation is always positive we can drop the absolute value bars and the condition to have the first branch becomes: $\sigma_1 = \sigma_2$.

The second branch of (9.11) is satisfied when the standard deviations are the same $\sigma_1 = \sigma_2$ and $\beta \neq 0$ (which implies that the means are different), in this case the equation will have a double solution that will be equal i.e. $x_1 = x_2 = \frac{\mu_1 - \mu_2}{2}$.

The third branch of the same Equation (9.11), represents the case when the two distributions have exactly the same means and standard deviations.

Finally, the last branch of Equation (9.11), is never possible, because that will imply that if $\beta = 0 \implies \mu_1 = \mu_2$ and $\gamma \neq 0 \implies \mu_1^2 \neq \mu_2^2$, which is impossible. Therefore, the original Equation (9.9) will always have two distinct, two equal or an infinite number of solutions.

By substitution we obtain the intersection points of the two Gaussian distributions $x_1 = 3.1573$ and respectively $x_2 = 7.5094$.

9.4.3. Evaluation on Modular Neural Networks

The current section will describe the following items in detail:

- The creation of a modular neural network system to be used for multi-class pattern recognition;
- The designed system's performance evaluation on the synthetic datasets;
- And finally, the designed system's performance evaluation on realistic datasets and comparison with other baseline classifiers.

The polynomial order of the decision boundary present in a 2-class dataset is predicted using the Meta-Measurement method and the best Statistical Method presented in previous sections. The order of the polynomial will be used as the number of hidden nodes in the architecture selection process for creating a Modular Neural Network (MNN). This MNN is then used to classify a dataset and the performance of such a MNN architecture will be assessed.

The outline of the steps involved in creating the Modular Neural Network (MNN) for pattern recognition are shown in Figures 9.15 and 9.16. The system diagram had to be split on two separate pages for readability.

In order to create and test a Modular Neural Network we have split the data available for the given pattern recognition problem into 3 sets, a training set, a validation set and a testing set. The training set is used to update the parameters of the neural network module. The validation

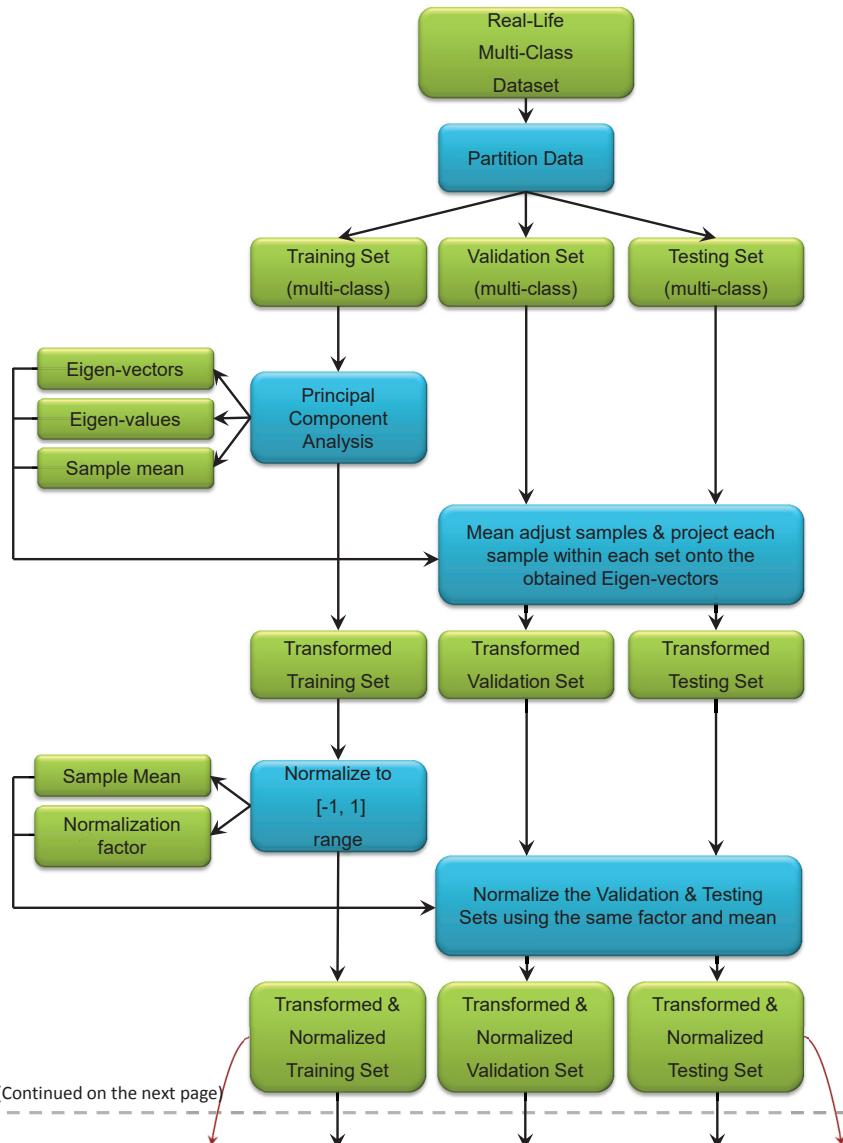


Fig. 9.15. Overview of the steps involved in assessing the classification performance of the MNN created using a number of hidden nodes suggested by the Meta-Measurement method (Continued on next page).

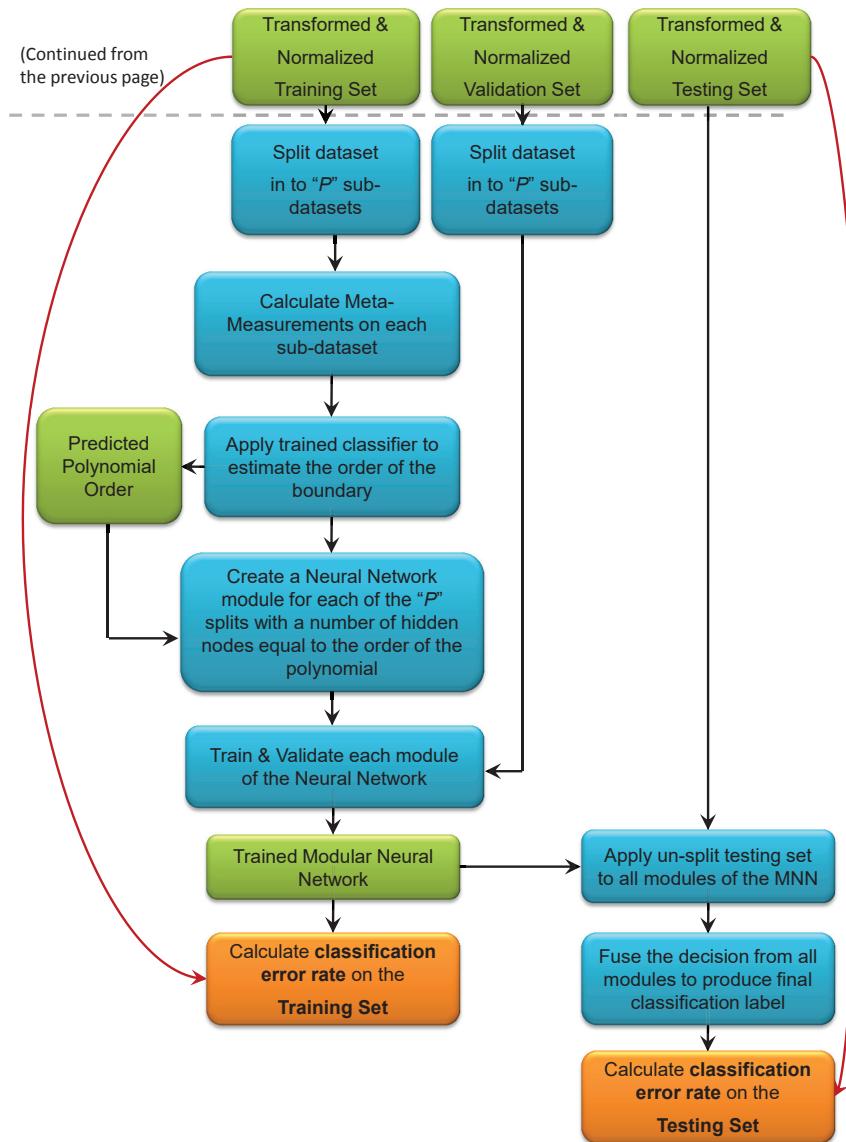


Fig. 9.16. Overview of the steps involved in assessing the classification performance of the MNN created using a number of hidden nodes suggested by the Meta-Measurement method (Continuation from the previous page).

set is used to determine at what epoch to stop updating the parameters of the neural network module. Finally, the testing set, is used exclusively as ground truth provider to calculate the accuracy of the prediction given by the MNN.

The steps involved in creating and testing the MNN are as follows:

- Partition the initial data available into three sets: training, validation and testing sets;
- Compute the eigen-values/vectors from the training set alone;
- Apply PCA to reduce the dimensionality of the dataset to 2 (this step is used only for the statistical methods to work);
- Normalize the scale of the resulting transformed datasets based on parameters estimated from the training set alone;
- Split the training and validation sets using a 1-versus-1 scheme into p sub-datasets;
- Create all the modules of the MNN by applying these steps for each of p sub-datasets;
 - (a) Obtain the meta-measurements or statistical polynomial order predictions for each sub-dataset;
 - (b) Create and train each a neural network module. Stop the training Mean Squared Error calculated on the validation set is increasing.

After completing this step training of the MNN has completed. Now the MNN is ready to be used in testing mode or in running mode. The testing procedure requires the following steps:

- Apply the testing set, that has not been split into multiple sub-datasets, but is in his raw form, to all of the modules in the MNN;
- Fuse the decisions from all the modules to form the output decision. This forms the class prediction that the MNN makes given the testing set.
- Compare the predictions made by the MNN to the labellings of the testing set. The result of this comparison is used as the classification error rate that will be reported.

It is very important to note that the training and validation sets are assumed to be known at the time of designing the MNN. But, the testing set is assumed to be unknown at the time of designing the system. Therefore only the training and validation sets are split into p 2-class sub-datasets.

We have taken extreme care in designing this system to reduce the biases introduced by improper handling of the data available for estimating the

parameters of the classifier system (training and validation sets) and not to be contaminated with the testing set.

Positive bias can be introduced in the performance evaluation of a system if we use all of the available data to obtain the eigen-vectors during a PCA feature reduction step. Also, a positive bias will be present in the evaluation if we use the whole dataset to obtain the normalization parameters (mean vectors and scaling factor vectors). In both cases, only the portion of the dataset that is reserved for training is used to obtain the eigen-vectors and normalization parameters.

We have clearly separated the steps taken to mitigate this positive bias as it can be seen in Figure 9.15. Another important detail that has to be mentioned is that the “Sample Mean” calculated during the PCA step is not the same as the “Sample Mean” that is used to normalize the data after the PCA step.

9.4.4. Results on Synthetic Data

We have assessed the performance of 1,000 MNNs, created using the procedure described in the previous section, on 1,000 synthetically generated datasets that are described in Section 9.3.1.

The number of hidden nodes corresponding to each NN module is suggested by one of the 5 prediction methods:

- Our own novel method using so called Meta-Measurements (denoted as META in subsequent figures);
- Statistical methods fit a polynomial of order 1 to 10 to the (denoted as STAT in subsequent figures);
- Oracle that is random guessing with a uniformly distributed probability the order from 1 to 10 (denoted as RAND in subsequent figures);
- Oracle based true order assignment obtained from the data generation step (denoted as TRUE in subsequent figures);
- Always predicting the maximum order 10 (denoted as ALL10 in subsequent figures).

The last three architecture prediction methods are included as a simple comparison to analyse the validity of our proposal.

We have analysed the performance of the MNN systems from 5 points of view.

- Firstly, we summarize the total number of parameters that were suggested by each of the 5 prediction methods in Figure 9.17 which shows

a bar chart with the total count of all the connection weights in all the generated NN modules.

The bar graph shows for each bar the actual total count of connection weights and the relative percentage variation considering the number of connections suggested by the TRUE Oracle as the base of comparison, which has a 100% percentage in its corresponding bar.

From this figure we can observe that the RAND, META and TRUE methods of suggesting the architecture of modules, produce roughly the same number of connection weights, about 23,000 and relative percentage close to 100%.

The other two prediction methods (STAT and ALL10) are overestimating the number of parameters. Obviously, the ALL10 method overestimates grossly the number of parameters by always using the most complicated NN module regardless of the complexity of the dataset. This method also provides the upper limit to the number of parameters.

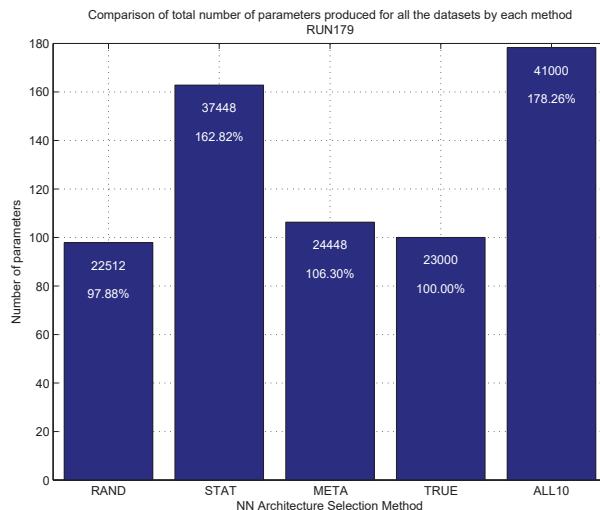


Fig. 9.17. Total number of parameters for all the 1000 NNs trained and assessed with each architecture prediction method.

- The second type of analysis we have done is to create, for each architecture suggestion method, a histograms showing the number of NNs achieving one of the following performance evaluation measures:
 - (a) classification performance, see Figure 9.18; The horizontal axis shows the bin centres of the histogram while vertical axis shows the number of NN modules to fall in each bin. The bins have following centre

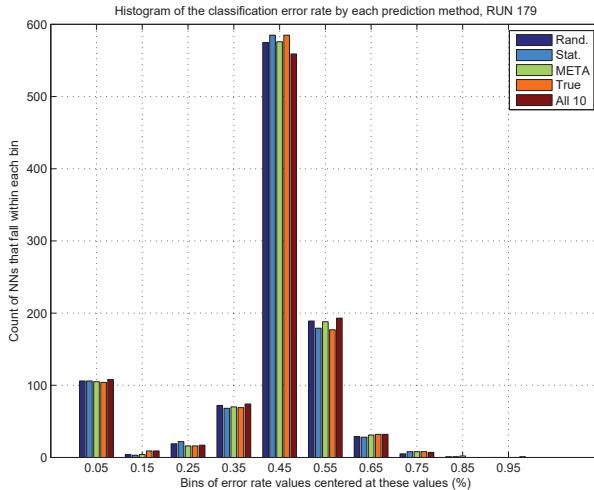


Fig. 9.18. Histogram of the number of epochs needed to meet the stopping criterion.

locations: While the bin edges are the following:

- (b) best achieved training epoch, see Figure 9.19; The horizontal axis shows the bin centres of the histogram while vertical axis shows the number of NN modules to fall in each bin. The bins have following centre location: While the bin edges are the following:

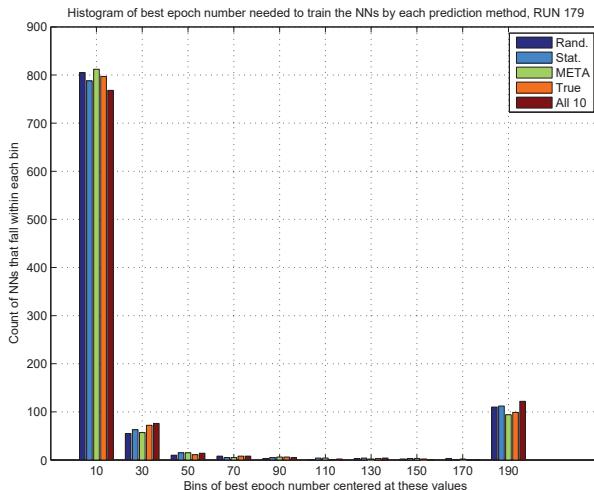


Fig. 9.19. Histogram of the training time of the NNs in our experiments.

- (c) best achieved training time, see Figure 9.20. The horizontal axis shows the bin centres of the histogram while vertical axis shows the number of NN modules to fall in each bin.

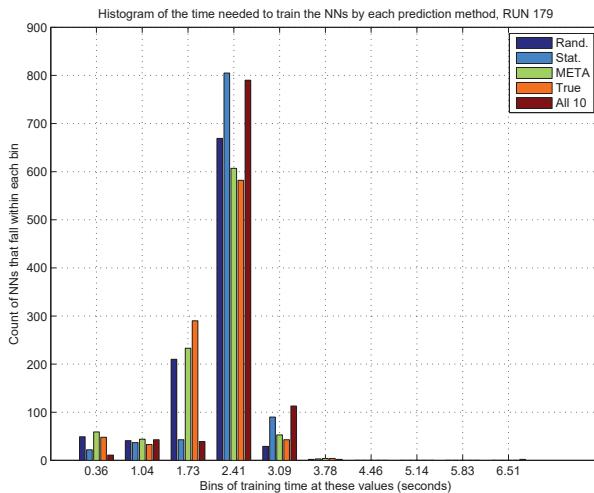


Fig. 9.20. Histogram of the NNs training time distribution.

- The analysis of the average error gradients for all 1,000 MNN modules suggests that the 5 architecture prediction methods seem to be learning the problems at the same rate.

From Figure 9.18 we can observe a common trend for all of the 5 architecture suggestion methods, they all seem to perform in the same fashion regardless of the chosen architecture.

The same is true for the best epoch reached, as it can be seen from Figure 9.19.

Only slight variations are to be observed among the 5 architecture prediction methods.

However, when we examine the training time required for the individual NN modules we observe that the ALL10 method produces a slightly larger number of NN that need a time between 2.74 seconds & 3.42 seconds to train.

9.5. Conclusions

Our contributions in this chapter are summarized as follows:

- A machine learning method, called ‘Meta-Measurement’, was developed to estimate the polynomial order of the decision boundary between the data points of a two class pattern recognition problem.
- We have proposed a method of improving the learning rate of neural networks by adapting the initial weights of the network to fit the polynomial functions that can be superimposed on the decision boundary found in the training set by exploiting the Meta-Measurement method above.
- Modular Neural Networks were constructed, whereby each module has the size (number of hidden nodes) suggested by the Meta-Measurement method. This method was compared to statistical methods of goodness of fit and also compared with 3 Oracle-based methods. These Oracles were either randomly guessing the correct order of the polynomial, or the true order of the polynomial, or finally the third Oracle was always suggesting the maximum order possible. Meta-Measurement based methodology resulted in networks with significantly fewer hidden nodes and performances comparable to benchmark classifiers.

Meta-Learning, architecture selection and classifier selection are fields that can benefit from great improvement. We have limited our scope to associating polynomial functions to decision boundaries. But other functions lend themselves to be investigated. For example the choices available for model selection can include Bezier curves and trigonometric functions.

References

- [1] Plato, *Republic : 1-2.368c4 / Plato ; with an introduction and commentary by Chris Emyln Jones*. Warminster : Aris & Phillips (2007). Ancient Greek and English text on facing pages.
- [2] Aristotle, *Physics / Translated with commentaries and glossary by Hippocrates G. Apostle*. Bloomington : Indiana University Press (1969).
- [3] N. J. Nilsson, *The quest for artificial intelligence: a history of ideas and achievements*. Cambridge University Press (2010).
- [4] T. G. Dietterich, Approximate statistical tests for comparing supervised classification learning algorithms, *Neural Computation*. **10**(7), pp. 1895–1923 (1998).
- [5] L. I. Kuncheva, *Combining Pattern Classifiers: Methods and Algorithms*. A Wiley-Interscience publication (2004). ISBN 0-471-21078-1 (cloth).

- [6] S. S. Haykin, *Neural networks and learning machines*, 3rd international edn. Harlow; London: Pearson Education (2009).
- [7] K.-L. Du and M. N. S. Swamy, *Neural networks and statistical learning*. Springer London (2014).
- [8] T. Poggio and F. Girosi, Networks for approximation and learning, *Proceedings of the IEEE*. **78**(9), 1481–1497 (1990).
- [9] A. N. Tikhonov, On solving incorrectly posed problems and method of regularization, *Doklady Akademii Nauk USSR*. **151**, 501–504 (1963).
- [10] K. Mehrotra, C. K. Mohan, and S. Ranka, *Elements of Artificial Neural Networks*, 2nd Printing. MIT Press (2000).
- [11] G. M. Foody, Using prior knowledge in artificial neural network classification with a minimal training set, *International Journal of Remote Sensing*. **16** (2), 301–312 (1995).
- [12] P. A. Blamire, The influence of relative sample size in training artificial neural networks, *International Journal of Remote Sensing*. **17**(1), 223–230 (1996).
- [13] M. Pal and P. M. Mather, An assessment of effectiveness of decision tree methods for land cover classification, *Remote Sensing of Environment*. **86** (4), 554–565 (2003).
- [14] T. Kavzoglu, Increasing the accuracy of neural network classification using refined training data, *Environmental Modelling & Software*. **24**(7), 850–858 (2009).
- [15] B. Widrow and M. A. Lehr, 30 years of adaptive neural networks: Perceptrons, MADALINE and backpropagation, *Proceedings of IEEE*. **78**(9), pp. 1415–1442 (1990).
- [16] E. Ronco and P. Gawthrop. Modular neural networks: a state of the art. Technical report: Csc-95026 (may 12, 1995), Centre for System and Control, University of Glasgow UK (1995).
- [17] G. Auda and M. Kamel, Modular neural networks: A survey, *International Journal of Neural Systems*. **9**(2), 129–151 (1999).
- [18] B.-L. Lu and M. Ito, Task decomposition and module combination based on class relations: A modular neural network for pattern classification, *IEEE Transactions on Neural Networks*. **10**(5) (1999).
- [19] D. A. Mitzias and B. G. Mertzios, A neural multiclassifier system for object recognition in robotic vision applications, *Science Direct, Measurement*. **36**, 315–330 (2004).
- [20] P. Poirazi, C. Neocleous, and et al., Classification capacity of a modular neural network implementing neurally inspired architecture and training rules, *IEEE Transactions on Neural Networks*. **15**(13) (2004).
- [21] T. K. Ho and M. Basu, Complexity measures of supervised classification problems, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **24**(3), pp. 289 – 300 (2002).
- [22] M. Basu and T. K. Ho, *Data Complexity in Pattern Recognition*. Springer (Advanced Information and Knowledge Processing) (2006).
- [23] J.-R. Cano, Analysis of data complexity measures for classification, *International Journal of Expert Systems with Applications*. **40**, pp. 4820–4831 (2013).

- [24] G. D. C. Cavalcanti, T. I. Ren, and B. A. Vale, Data complexity measures and nearest neighbor classifiers: A practical analysis for meta-learning, *IEEE 24th International Conference on Tools with Artificial Intelligence.* **1**, pp. 1065–1069 (2012). doi: 10.1109/ICTAI.2012.150.
- [25] J. M. Sotoca, R. A. Mollineda, and J. S. Sánchez, A meta-learning framework for pattern classification by means of data complexity measures, *Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial.* **29**, pp. 31–38 (2006). URL [http://www.aepia.org/revista\(c\)AEPIA](http://www.aepia.org/revista(c)AEPIA).
- [26] B. G. Gherman and K. Sirlantzis, Polynomial order prediction using a classifier trained on meta-measurements, *4th International Conference on Emerging Security Technologies.* pp. pp. 117–120 (2013). doi: 10.1109/EST.2013.26.
- [27] D. A. Levin, Y. Peres, and E. L. Wilmer, *Markov Chains and Mixing Times*. American Mathematical Society (2009).

Chapter 10

Multi-Criterion Optimization and Decision Making Using Evolutionary Computing

Kalyanmoy Deb

*Department of Electrical and Computer Engineering
Michigan State University, East Lansing, MI, USA*

kdeb@egr.msu.edu

Most problem solving tasks including pattern recognition activities involve more than one conflicting objectives, such as accuracy, execution time, and generalization ability of a pattern matching algorithm. In such problems, there exist not one, but a number of optimal trade-off solutions. In this chapter, we discuss a number of classical and evolutionary multi-criterion optimization methods. Although no results on any pattern recognition problem has been presented, the methodologies and their scope in other problem solving tasks do not leave with much doubt about their use in different practical problem solving tasks.

10.1. Introduction

Multi-criterion optimization problems involve multiple conflicting objectives, which are to be simultaneously minimized or maximized. Since each of the criterion corresponds to a different optimal solution, the solution to such problems is not one, but a number of trade-off optimal solutions. The optimal set includes individual optimal solutions to each criterion and also a number of compromise solutions trading-off the considered criteria. Thus, multi-criterion optimization problem involves two main tasks:

- (1) Multi-criterion optimization (MO), in which a number of Pareto-optimal solutions are sought, and
- (2) Multi-criterion decision-making (MCDM), in which a single preferred solution is chosen from the Pareto-optimal set.

The first task can be automated once the problem description is clearly defined, whereas the second task must involve human decision-makers and

certain MCDM methods. Although it may be construed from the above that the second task is dependent on the first task, these two tasks can be applied in any order. Traditionally, the second task was executed first in order to determine scalarization schemes for converting multiple criteria into a single composite criterion. Most classical methods, such as the weighted-sum method, the epsilon-constraint method, etc. [1–3], utilize this concept. The single composite criterion is then optimized and the resulting solution is declared as the final solution.

More recent generational MCDM methods [2] and evolutionary multi-criterion optimization (EMO) [4, 5] methods use the optimization first and then a MCDM approach is used to choose a single preferred solution. However, most recent interactive EMO-MCDM methods [6, 7] use MCDM approaches within an EMO procedure and interactively with a human decision-makers. These interactive methods are promising for addressing problems with a large number of criteria.

In the remainder of this chapter, we present the basics of multi-criterion optimization in Section 10.2. Thereafter, we present two popularly used classical multi-criterion optimization methods in Section 10.3. Section 10.4 discusses evolutionary multi-criterion optimization methods in detail and present a popular method – elitist non-dominated sorting genetic algorithm or NSGA-II [8]. Thereafter, in Section 10.5, results using normal constraint method and NSGA-II are compared. Different combined EMO-MCDM approaches are described next in Section 10.6. Section 10.7 presents a number of advanced topics in multi-criterion optimization. Finally, Section 10.8 concludes the chapter.

10.2. Principles of Multi-Criterion Optimization

A multi-objective optimization problem involves a number of objective functions which are to be either minimized or maximized subject to a number of constraints and variable bounds:

$$\begin{aligned}
 & \text{Minimize/Maximize } f_m(\mathbf{x}), & m = 1, 2, \dots, M; \\
 & \text{Subject to } g_j(\mathbf{x}) \geq 0, & j = 1, 2, \dots, J; \\
 & h_k(\mathbf{x}) = 0, & k = 1, 2, \dots, K; \\
 & x_i^{(L)} \leq x_i \leq x_i^{(U)}, & i = 1, 2, \dots, n.
 \end{aligned} \tag{10.1}$$

A solution $\mathbf{x} \in \mathbf{R}^n$ is a vector of n decision variables: $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$. The solutions satisfying the constraints and variable bounds constitute a

Feasible set S in the decision variable space \mathbf{R}^n . One of the striking differences between single-objective and multi-objective optimization is that in multi-objective optimization the objective function vectors belong to a multi-dimensional objective space \mathbf{R}^M . The objective function vectors constitute a feasible set Z in the objective space. For each solution \mathbf{x} in S , there exists a point $\mathbf{z} \in Z$, denoted by $\mathbf{f}(\mathbf{x}) = \mathbf{z} = (z_1, z_2, \dots, z_M)^T$. To make the descriptions clear, we refer a decision variable vector as a solution and the corresponding objective vector as a point.

The optimal solutions in multi-objective optimization can be defined from a mathematical concept of *partial ordering* [9]. In the parlance of multi-objective optimization, the term *domination* is used for this purpose. In this section, we restrict ourselves to discuss unconstrained (without any equality, inequality or bound constraints) optimization problems. The domination between two solutions is defined as follows [2, 4]:

Definition 10.1. A solution $\mathbf{x}^{(1)}$ is said to dominate the another solution $\mathbf{x}^{(2)}$, if both the following conditions are true:

- (1) The solution $\mathbf{x}^{(1)}$ is no worse than $\mathbf{x}^{(2)}$ in all objectives. Thus, the solutions are compared based on their objective function values (or location of the corresponding points $(\mathbf{z}^{(1)} \text{ and } \mathbf{z}^{(2)})$ in the objective function set Z).
- (2) The solution $\mathbf{x}^{(1)}$ is strictly better than $\mathbf{x}^{(2)}$ in at least one objective.

For a given set of solutions (or corresponding points in the objective function set Z , for example, those shown in Figure 10.1(a)), a pair-wise comparison can be made using the above definition and whether one point dominates another point can be established.

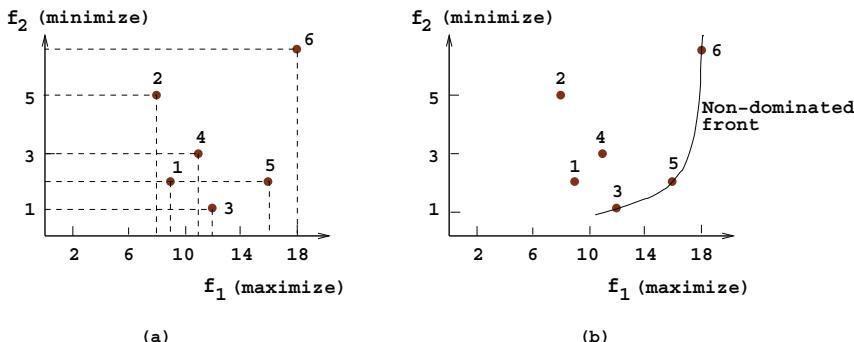


Fig. 10.1. A set of points and the first non-dominated front are shown.

All points which are not dominated by any other member of the set are called the non-dominated points of class one, or simply the non-dominated points. For the set of six points shown in the figure, they are points 3, 5, and 6. One property of any two such points is that a gain in an objective from one point to the other happens only due to a sacrifice in at least one other objective. This *trade-off* property between the non-dominated points makes the practitioners interested in finding a wide variety of them before making a final choice. These points make up a front when viewed together on the objective space; hence the non-dominated points are often visualized to represent a *non-dominated front*. The theoretical computational effort needed to select the points of the non-dominated front from a set of N points is $O(N \log N)$ for 2 and 3 objectives, and $O(N \log^{M-2} N)$ for $M > 3$ objectives [10], but for a moderate number of objectives, the procedure need not be particularly computationally efficient in practice.

With the above concept, now it is easier to define the *Pareto-optimal solutions* in a multi-objective optimization problem. If the given set of points for the above task contain *all* feasible points in the objective space, the points lying on the first non-domination front, by definition, do not get dominated by any other point in the objective space, hence are Pareto-optimal points (together they constitute the Pareto-optimal front) and the corresponding pre-images (decision variable vectors) are called Pareto-optimal solutions. However, more mathematically elegant definitions of Pareto-optimality (including the ones for continuous search space problems) exist in the multi-objective optimization literature [2, 11]. Interested readers are encouraged to refer to these references.

Obviously, the above definition and procedure of arriving at Pareto-optimal solutions is not a practical approach, as it involves finding all solutions in the search space. According to no-free-lunch theorem [12], since no single mathematical or classical optimization algorithm exists that would solve all single-objective optimization problems efficiently, the no-free-lunch theorem can also be extended for multi-objective optimization and a similar conclusion can be made [13]. Therefore, in solving arbitrary multi-objective optimization problems, our goal is use an efficient algorithm that would reach close to the true Pareto-optimal solutions. In the next two sections, we present classical and evolutionary optimization algorithms that in most problems consider only a tiny fraction search space and proceed near the Pareto-optimal solutions with iterations.

10.3. Classical Multi-Criterion Optimization Methods

In this section, we briefly mention two commonly-used classical multi-objective optimization methods.

10.3.1. Weighted-Sum Approach

The weighted sum method, as the name suggests, scalarizes a set of objectives into a single objective by pre-multiplying each objective with a user-supplied weight. This method is the simplest approach and is probably the most widely used classical approach. If we are faced with the two objectives of minimizing the cost of a product and minimizing the amount of wasted material in the process of fabricating the product, one naturally thinks of minimizing a weighted sum of these two objectives. Although the idea is simple, it introduces a not-so-simple question. What values of the weights must one use? Of course, there is no unique answer to this question. The answer depends on the importance of each objective in the context of the problem and a scaling factor. The scaling effect can be avoided somewhat by normalizing the objective functions. After the objectives are normalized, a composite objective function $F(\mathbf{x})$ can be formed by summing the weighted normalized objectives and the problem is then converted to a single-objective optimization problem as follows:

$$\begin{aligned} \text{Minimize } F(\mathbf{x}) &= \sum_{m=1}^M w_m f_m(\mathbf{x}), \\ \text{Subject to } g_j(\mathbf{x}) &\geq 0, \quad j = 1, 2, \dots, J; \\ x_i^{(L)} &\leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \dots, n. \end{aligned} \quad (10.2)$$

Here, w_m ($\in [0, 1]$) is the weight of the m -th objective function. Since the minimum of the above problem does not change if all weights are multiplied by a constant, it is the usual practice to choose weights such that their sum is one, or $\sum_{m=1}^M w_m = 1$.

Mathematically oriented readers may find a number of interesting theorems regarding the relationship between the optimal solution of the above problem to the true Pareto-optimal solutions in classical texts [1, 2, 14].

Let us now illustrate how the weighted sum approach can find Pareto-optimal solutions of the original problem. For simplicity, we consider the two-objective problem shown in Figure 10.2. The feasible objective space and the corresponding Pareto-optimal solution set are shown. With two objectives, there are two weights w_1 and w_2 , but only one is independent. Knowing any one, the other can be calculated by simple subtraction.

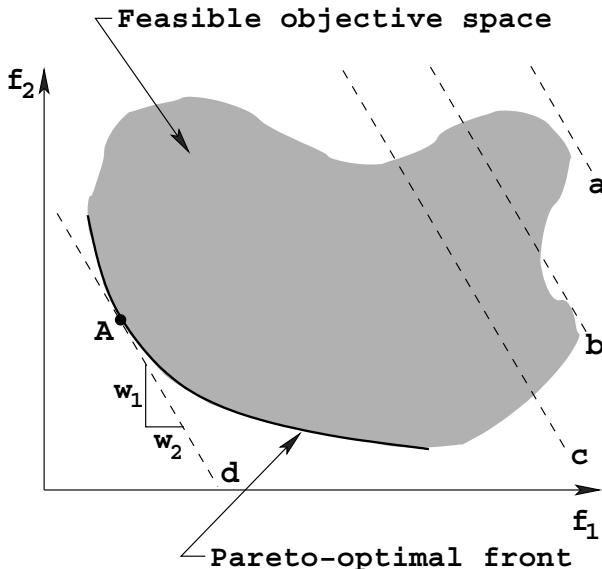


Fig. 10.2. Illustration of the weighted sum approach on a convex Pareto-optimal front.

It is clear from the figure that a choice of a weight vector corresponds to a pre-destined optimal solution on the Pareto-optimal front, as marked by the point A. By changing the weight vector, a different Pareto-optimal point can be obtained. However, there are a couple of difficulties with this approach:

- (1) A uniform choice of weight vectors do not necessarily find a uniform set of Pareto-optimal solutions on the Pareto-optimal front [4].
- (2) The procedure cannot be used to find Pareto-optimal solutions which lie on the non-convex portion of the Pareto-optimal front.

The former issue makes it difficult for the weighted-sum approach to be applied reliably to any problem in order to find a good representative set of Pareto-optimal solutions. The latter issue arises due to the fact a solution lying on the non-convex Pareto-optimal front can never be the optimal solution of problem given in Equation 10.2.

10.3.2. ϵ -Constraint Method

In order to alleviate the difficulties faced by the weighted sum approach in solving problems having non-convex objective spaces, the ϵ -constraint

method is used. Haimes *et al.* [15] suggested reformulating the multi-objective optimization problem by just keeping one of the objectives and restricting the rest of the objectives within user-specified values. The modified problem is as follows:

$$\begin{aligned} & \text{Minimize } f_\mu(\mathbf{x}), \\ & \text{Subject to } f_m(\mathbf{x}) \leq \epsilon_m, \quad m = 1, 2, \dots, M \text{ and } m \neq \mu; \\ & \quad g_j(\mathbf{x}) \geq 0, \quad j = 1, 2, \dots; \\ & \quad h_k(\mathbf{x}) = 0, \quad k = 1, 2, \dots, K; \\ & \quad x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \dots, n. \end{aligned} \quad (10.3)$$

In the above formulation, the parameter ϵ_m represents an upper bound of the value of f_m and need not necessarily mean a small value close to zero.

Let us say that we retain f_2 as an objective and treat f_1 as a constraint: $f_1(\mathbf{x}) \leq \epsilon_1$. Figure 10.3 shows four scenarios with different ϵ_1 values.

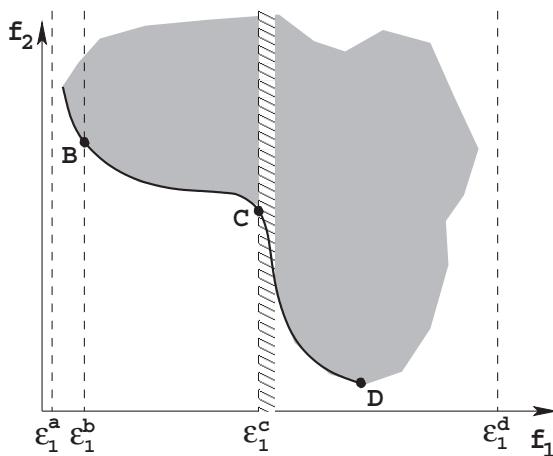


Fig. 10.3. The ϵ -constraint method.

Let us consider the third scenario with $\epsilon_1 = \epsilon_1^c$ first. The resulting problem with this constraint divides the original feasible objective space into two portions, $f_1 \leq \epsilon_1^c$ and $f_1 > \epsilon_1^c$. The left portion becomes the feasible solution of the resulting problem stated in Equation (10.3). Now, the task of the resulting problem is to find the solution which minimizes this feasible region. From Figure 10.3, it is clear that the minimum solution is ‘C’. In this way, intermediate Pareto-optimal solutions can be obtained in

the case of non-convex objective space problems by using the ϵ -constraint method.

One of the difficulties of this method is that the solution to the problem stated in Equation (10.3) largely depends on the chosen ϵ vector. Let us refer to Figure 10.3 again. Instead of choosing ϵ_1^c , if ϵ_1^a is chosen, there exists no feasible solution to the stated problem. Thus, no solution would be found. On the other hand, if ϵ_1^d is used, the entire search space is feasible. The resulting problem has the minimum at ‘D’. Moreover, as the number of objectives increases, there exist more elements in the ϵ vector, thereby requiring more information from the user.

10.3.3. Normal Constraint Method (NC)

A smart extension of the above ϵ -constraint method is described next. Instead of using axis-parallel constraints, inclined constrained hyper-planes are used. The steps are discussed below.

First, each objective function $f_i(\mathbf{x})$ with all given constraints and variable bounds is optimized individually. Let us assume that the respective solution is \mathbf{x}^i . At this point, all objective function values are computed and the objective vector \mathbf{f}^i is computed. Figure 10.4 shows points A and B as \mathbf{f}^1 and \mathbf{f}^2 for objective functions f_1 and f_2 , respectively.

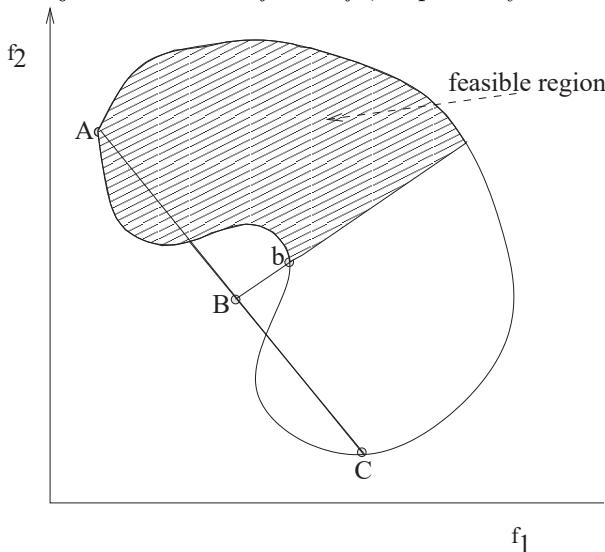


Fig. 10.4. Schematic showing obtained solutions in NC method.

Note that \mathbf{f}^i may lie on the weak part of the Pareto-optimal set, if exists in a problem. Second, M individual objective vectors are used to create a $(M-1)$ -dimensional hyperplane, shown as line AB in the figure. Thereafter, in third step, $(M-1)$ direction vectors ($N_i = \mathbf{f}^M - \mathbf{f}^i$, $i = 1, 2, \dots, (M-1)$) are created connecting each of the $(M-1)$ individual objective points \mathbf{f}^i ($i = 1, 2, \dots, (M-1)$) with \mathbf{f}^M . Also, uniformly distributed set of points (\mathbf{Z}_β) on the hyper-plane are systematically chosen [16] and objective vectors that make an acute angle with each of the $(M-1)$ direction vectors are considered feasible. The following optimization problem is then solved at each \mathbf{Z}_β point:

$$\begin{aligned} & \text{Minimize } f_M(\mathbf{x}), \\ & \text{Subject to } N_i^T (\mathbf{f}(\mathbf{x}) - \mathbf{Z}_\beta) \leq 0, \text{ for } i = 1, 2, \dots, (M-1), \\ & \quad \mathbf{x} \in \mathcal{S}, \end{aligned} \quad (10.4)$$

where \mathcal{S} is the feasible set. Figure 10.4 shows the feasible objective space for a two-objective optimization problems with $\mathbf{Z}_\beta = \mathbf{B}$. The resulting solution of the above problem is the point b, as shown in the figure. Thus, by changing the point B along the line AB, different Pareto-optimal points can be generated. It is interesting to realize that not all points obtained by NC method is guaranteed to be Pareto-optimal.

10.3.4. Critical Comments on Point-based MO Methods

Having discussed a few classical point-by-point methods for multi-objective optimization, we now criically discuss the basic principle of these approaches. Consider Figure 10.5, in which we sketch how multiple independent parametric single-objective optimization applications may find different Pareto-optimal solutions using the so-called *generational* methods.

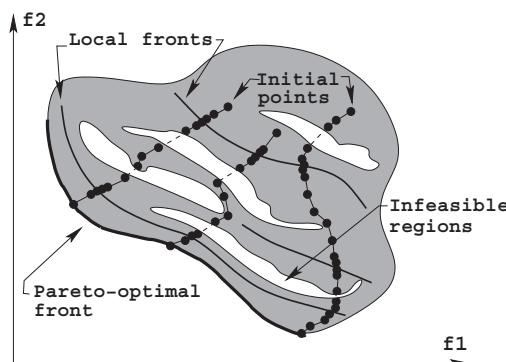


Fig. 10.5. Principle of generational multi-objective optimization method employing multiple independent single-objective optimizations.

It is worth highlighting here that the Pareto-optimal front corresponds to global optimal solutions of several problems each formed with a different scalarization of objectives. During the course of an optimization task, algorithms must overcome a number of difficulties, such as infeasible regions, local optimal solutions, flat or non-improving regions of objective landscapes, isolation of optimum, etc., to finally converge to the global optimal solution. Moreover, due to practical limitations, an optimization task must also be completed in a reasonable computational time. All these difficulties in a problem require that an optimization algorithm strikes a good balance between *exploring* new search directions and *exploiting* the extent of search in currently-best search direction. When multiple runs of an algorithm need to be performed independently to find a set of Pareto-optimal solutions, the above balancing act must have to be performed not only in one, but in every single run. Since runs are performed independently from one another, no information about the success or failure of previous runs is utilized to speed up the overall process. In difficult multi-objective optimization problems, such memory-less, generational methods may demand a large overall computational overhead to find a set of Pareto-optimal solutions [17], which we demonstrate in Section 10.5. Moreover, despite the issue of global convergence, independent runs may not guarantee achieving a good distribution among obtained points by an easy variation of scalarization parameters.

We now describe a completely different optimization method which works with a population of points in each generation, thereby allowing it to be used to find multiple trade-off solutions in a single simulation run.

10.4. Evolutionary Multi-Criterion Optimization Methods

Over the past two decades, researchers have suggested a number of different evolutionary multi-criterion methods (EMO) emphasizing non-dominated solutions in a EA population. In this section, we shall describe one state-of-the-art algorithm popularly used in EMO studies.

10.4.1. *Elitist Non-Dominated Sorting GA (NSGA-II)*

The non-dominated sorting GA or NSGA-II procedure [8] for finding multiple Pareto-optimal solutions in a multi-objective optimization problem has the following three features:

- (1) It uses an elitist principle,
- (2) It uses an explicit diversity preserving mechanism, and
- (3) It emphasizes the non-dominated solutions.

In NSGA-II, the offspring population Q_t is first created by using the parent population P_t and the usual genetic operators [18]. Thereafter, the two populations are combined together to form R_t of size $2N$. Then, a non-dominated sorting is used to classify the entire population R_t . Once the non-dominated sorting is over, the new population is filled by solutions of different non-dominated fronts, one at a time. The filling starts with the best non-dominated front and continues with solutions of the second non-dominated front, followed by the third non-dominated front, and so on. Since the overall population size of R_t is $2N$, not all fronts may be accommodated in N slots available in the new population. All fronts which could not be accommodated are simply deleted. When the last allowed front is being considered, there may exist more solutions in the last front than the remaining slots in the new population. This scenario is illustrated in Figure 10.6. Instead of arbitrarily discarding some members from the last acceptable front, the solutions which will make the diversity of the selected solutions the highest are chosen.

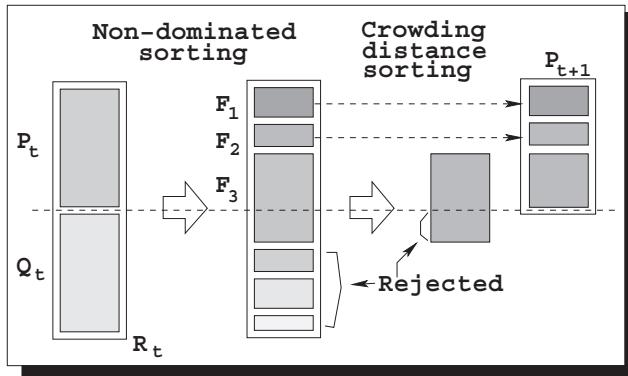


Fig. 10.6. Schematic of the NSGA-II procedure.

The NSGA-II procedure is outlined in the following.

Algorithm 10.1. NSGA-II

Step 1 Combine parent and offspring populations and create $R_t = P_t \cup Q_t$.
Perform a non-dominated sorting to R_t and identify different fronts: \mathcal{F}_i , $i = 1, 2, \dots$, etc.

Step 2 Set new population $P_{t+1} = \emptyset$. Set a counter $i = 1$.

Until $|P_{t+1}| + |\mathcal{F}_i| < N$, perform $P_{t+1} = P_{t+1} \cup \mathcal{F}_i$ and $i = i + 1$.

Step 3 Perform the $\text{Crowding-sort}(\mathcal{F}_i, <_c)$ procedure and include the most widely spread ($N - |P_{t+1}|$) solutions by using the crowding distance values in the sorted \mathcal{F}_i to P_{t+1} .

Step 4 Create offspring population Q_{t+1} from P_{t+1} by using the crowded tournament selection, crossover and mutation operators.

In Step 3, the crowding-sorting of the solutions of front i (the last front which could not be accommodated fully) is performed by using a *crowding distance metric*, which we describe a little later. The population is arranged in a descending order of magnitude of the crowding distance values. In Step 4, a crowding tournament selection operator, which also uses the crowding distance, is used.

The crowded comparison operator ($<_c$) compares two solutions and returns the winner of the tournament. It assumes that every solution i has two attributes:

- (1) A non-domination rank r_i in the population.
- (2) A local Crowding distance (d_i) in the population.

The crowding distance d_i of a solution i is a measure of the search space around i which is not occupied by any other solution in the population. Based on these two attributes, we can define the crowded tournament selection operator as follows.

Definition 10.2. *Crowded Tournament Selection Operator:* A solution i wins a tournament with another solution j if any of the following conditions are true:

- (1) If solution i has a better rank, that is, $r_i < r_j$.
- (2) If they have the same rank but solution i has a better crowding distance than solution j , that is, $r_i = r_j$ and $d_i > d_j$.

The first condition makes sure that chosen solution lies on a better non-dominated front. The second condition resolves the tie of both solutions being on the same non-dominated front by deciding on their crowded distance. The one residing in a less crowded area (with a larger crowding distance d_i) wins. The crowding distance d_i can be computed in various ways. However, in NSGA-II, we use a crowding distance metric, which requires $O(MN \log N)$ computations.

To get an estimate of the density of solutions surrounding a particular solution i in the population, we take the average distance of two solutions

on either side of solution i along each of the objectives. This quantity d_i serves as an estimate of the perimeter of the cuboid formed by using the nearest neighbors as the vertices (we call this the *crowding distance*). In Figure 10.7, the crowding distance of the i -th solution in its front (marked with filled circles) is the average side-length of the cuboid (shown by a dashed box).

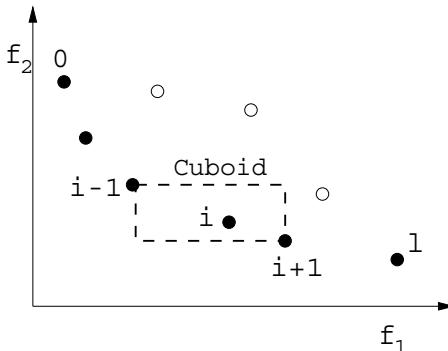


Fig. 10.7. The crowding distance calculation.

The following algorithm is used to calculate the crowding distance of each point in the set \mathcal{F} .

Algorithm 10.2. Crowding Distance Assignment Procedure:

Crowding-sort($\mathcal{F}, <_c$)

Step C1 Call the number of solutions in \mathcal{F} as $l = |\mathcal{F}|$. For each i in the set, first assign $d_i = 0$.

Step C2 For each objective function $m = 1, 2, \dots, M$, sort the set in worse order of f_m or, find the sorted indices vector: $I^m = \text{sort}(f_m, >)$.

Step C3 For $m = 1, 2, \dots, M$, assign a large distance to the boundary solutions, or $d_{I_1^m} = d_{I_l^m} = \infty$, and for all other solutions $j = 2$ to $(l - 1)$, assign:

$$d_{I_j^m} = d_{I_j^m} + \frac{f_m^{(I_{j+1}^m)} - f_m^{(I_{j-1}^m)}}{f_m^{\max} - f_m^{\min}}.$$

The index I_j denotes the solution index of the j -th member in the sorted list. Thus, for any objective, I_1 and I_l denote the lowest and highest objective function values, respectively. The second term on the right side of the last equation is the difference in objective function values between

two neighboring solutions on either side of solution I_j . Thus, this metric denotes half of the perimeter of the enclosing cuboid with the nearest neighboring solutions placed on the vertices of the cuboid (Figure 10.7). It is interesting to note that for any solution i the same two solutions $(i+1)$ and $(i-1)$ need not be neighbors in all objectives, particularly for $M \geq 3$. The parameters f_m^{\max} and f_m^{\min} can be set as the population-maximum and population-minimum values of the m -th objective function. The above metric requires M sorting calculations in Step C2, each requiring $O(N \log N)$ computations. Step C3 requires N computations. Thus, the complexity of the above distance metric computation is $O(MN \log N)$ and the overall complexity of one generation of NSGA-II is $O(MN^2)$, governed by the non-dominated sorting procedure.

It is clear from the above discussion that EMO constitutes an inherent parallel search. When a particular population member overcomes certain difficulties and makes a progress towards the Pareto-optimal front, its variable values and their combination must reflect this fact. When a recombination takes place between this solution and another population member, such valuable information of variable value combinations gets shared through variable exchanges and blending, thereby making the overall task of finding multiple trade-off solutions a parallelly processed task. We now compare classical and evolutionary methods on a number of multi-objective optimization problems.

10.5. Results of Multi-Criterion Optimization Methods

First, we consider two two-objective ZDT test problems [4, 19]. The test problems are slightly modified so that they become unconstrained multi-objective optimization problems. Thereafter, we present results on a three-objective DTLZ problem [19]. Each NC subproblem is solved using matlab's fmincon routine. Identical solution evaluations of both NC method and NSGA-II are used to make a fair comparison.

The modified ZDT1 test problem is stated as follows:

$$\begin{aligned} & \text{Minimize } f_1(\mathbf{x}) = x_1, \\ & \text{Minimize } f_2(\mathbf{x}) = g(\mathbf{x}) \left(1 - \sqrt{\frac{x_1}{g(\mathbf{x})}} \right), \\ & \text{where } g(x) = 1 + \sum_{i=2}^n x_i^2, \end{aligned} \quad (10.5)$$

where the box constraints are $x_1 \in [0, 1]$, and $x_i \in [-1, 1]$ for $i = 2, 3, \dots, n$. Here, we choose $n = 30$. This modified ZDT1 problem has a convex Pareto-

optimal front for which solutions correspond to $0 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \dots, n$.

After individual objective values are optimized, following optimal solutions are obtained: $\mathbf{f}^1 = (0, 3.5)^T$ and $\mathbf{f}^2 = (1, 0)^T$. The presence of a weak Pareto-optimal front is clear. Figure 10.8 shows the NC-obtained points after 20,000 solution evaluations. The NC method cannot avoid finding weak Pareto-optimal points. However, as shown in Figure 10.9, NSGA-II with an identical number of solution evaluations is able to find a good distributed set of points on the non-weak part of the Pareto-optimal set.

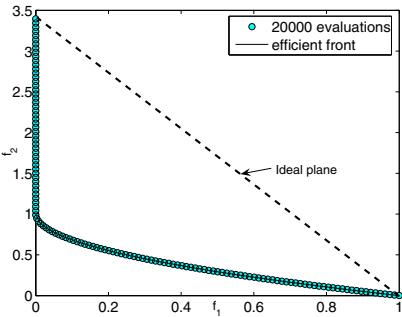


Fig. 10.8. Performance of NC method on ZDT1.

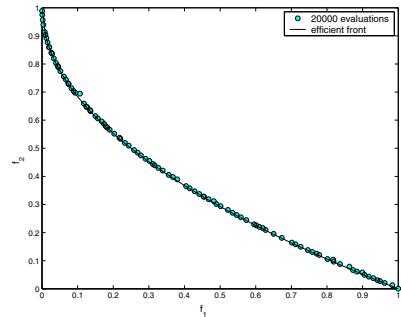


Fig. 10.9. Performance of NSGA-II method on ZDT1.

Next, we use the 10-variable ZDT4 test problem as it is suggested elsewhere [4]. This problem has a total of 100 distinct local efficient fronts in the objective space. The global Pareto-optimal solutions correspond to $0 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, 3, \dots, n$. The algorithms face a difficulty in overcoming a large number of local fronts and converging to the global front. Figure 10.10 shows the performance of the NC method after 100,000 evaluations. It also not able to reach the global Pareto-optimal front. However, Figure 10.11 shows that NSGA-II with only 20,000 evaluations is able to converge to the global efficient frontier.

The problem ZDT4 provides difficulty in terms of multi-modality of the search space. It is evident from the simulation results that the classical generating methods face enormous problems in overcoming the multi-modalities, whereas in this type of problems evolutionary multi-objective (EMO) methods are particularly found to be useful. This problem makes a clear distinction between population-based and point-by-point-based generating methods of solving multi-objective optimization problems. To con-

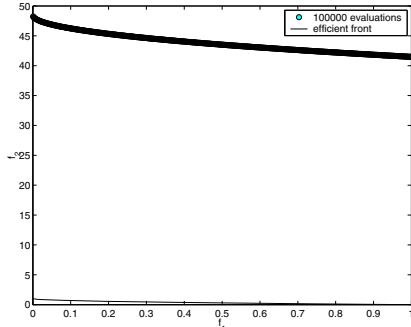


Fig. 10.10. Performance of NC method on ZDT4.

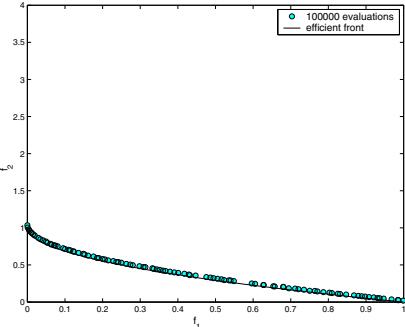


Fig. 10.11. Performance of NSGA-II method on ZDT4.

verge to the efficient frontier in this problem, an optimization algorithm must have to overcome a number of local efficient frontiers. Since in finding every Pareto-optimal solution, such hurdles have to be overcome, it is too demanding for a classical point-by-point approach to expect to do the task well independently every time. On the other hand, a population-based evolutionary approach recombines the decision variable vectors of good solutions to create new and hopefully better solutions through its recombination operator. Thus, if one population member somehow reaches close to the global efficient frontier, it can *pull* the rest of population members close to the global frontier by means of population interactions, thereby causing a faster and more reliable search.

Finally, we consider the 12-variable DTLZ2 test problem [19] having a spherical efficient front satisfying $f_1^2 + f_2^2 + f_3^2 = 1$ in the range $f_1, f_2 \in [0, 1]$. Figure 10.12 shows the performance of the NC approach on this problem. It is seen that the SQP strategy along with the NC method is not able to find most of Pareto-optimal solutions even after 100,000 solution evaluations. On the other hand, NSGA-II is able to find a reasonably well distributed set of points on the Pareto-optimal front.

The above three comparisons suggest that for simpler problems, the NC method performs similar to NSGA-II, but when the problem gets difficult or higher-dimensional, the NC method's serial application procedure cannot achieve comparable results with NSGA-II.

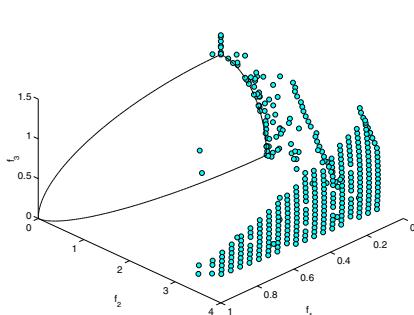


Fig. 10.12. Performance of NC method on DTLZ2 using 100,000 evaluations.

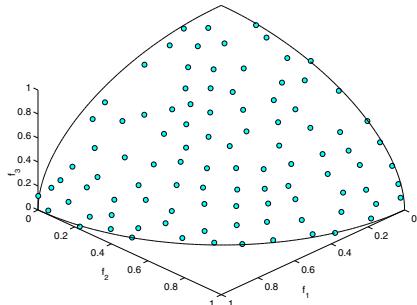


Fig. 10.13. Performance of clustered NSGA-II method on DTLZ2 using 100,000 evaluations.

10.6. Combined EMO and MCDM Methods

Searching for a set of Pareto-optimal solutions by using an EMO fulfills only one aspect of multi-objective optimization, as choosing a particular solution for an implementation is the remaining decision-making task which is equally important. For many years, EMO researchers have postponed the decision-making aspect and concentrated on developing efficient algorithms for finding multiple trade-off solutions. Having pursued that part somewhat, now for the past few years, EMO researchers are putting efforts to design combined algorithms for optimization and decision making. In the view of the author, the decision-making task can be combined with an EMO in the following three ways:

- (1) **A priori approach:** In this approach, a fixed decision making scheme is adopted before applying the optimization task. Classical weighted-sum, ϵ -constraint approach or other multi-criterion decision-making schemes can be used for this purpose [1, 2]. The decision-making information thus provided can then be used to solve the resulting single-objective optimization problem. Since the decision-making is called for without any knowledge of the trade-off choices, fixation of weights or ϵ -vectors or other parameters associated with the task becomes difficult and subjective.
- (2) **A posteriori approach:** In this approach, first an EMO is applied to find a set of well-distributed trade-off solutions across the Pareto-optimal front and then a decision-making aid is used to choose a single preferred solution. Since trade-off solutions are already available before

a decision-making aid is applied, this approach is practical and more meaningful than a priori approach. This is the principle with which EMO algorithms were developed. There are two ways the decision-making aids can be used:

- (a) **Generic consideration:** There are some aspects which most practical users would like to use in narrowing down their choice. We have discussed above the importance of finding robust and reliable solutions in the presence of uncertainties in decision variables and/or problem parameters. In such scenarios, an EMO methodology can straightway find a robust or a reliable frontier [20, 21] and no subjective preference from any decision maker may be necessary. Similarly, if a problem resorts to a Pareto-optimal front having *knee* points, such points are often the choice of decision makers. Knee points demands a large sacrifice in at least one objective to achieve a small gain in another thereby making it discouraging to move out from a knee point [22]. Other such generic choices are related to Pareto-optimal points depicting certain pre-specified relationship between objectives, Pareto-optimal points having multiplicity (say, at least two or more solutions in the decision variable space mapping to identical objective values), Pareto-optimal solutions which do not lie close to variable boundaries, Pareto-optimal points having certain mathematical properties, such as all Lagrange multipliers having more or less identical magnitude – a condition often desired to make an equal importance to all constraints, and others. These considerations are motivated from the fundamental and practical aspects of optimization and may be applied to most multi-objective problem solving tasks, without any consent of a decision-maker. These considerations may narrow down the set of non-dominated points. A further subjective consideration (discussed below) may then be used to pick a preferred solution.
- (b) **Subjective consideration:** In this category, any problem-specific information can be used to narrow down the choices and the process may even lead to a single preferred solution at the end. Most decision-making procedures use some preference information (utility functions, Reference points [23], Reference directions [24], marginal rate of return and a host of other considerations [2]) to select a subset of Pareto-optimal solutions. A recent book [25] is dedicated to the discussion of many such multi-criteria decision analysis (MCDA)

tools and collaborative suggestions of using EMO with such MCDA tools. Some hybrid EMO and MCDA algorithms are suggested in the recent past [26–30].

Many other generic and subjective considerations are needed and it is interesting that EMO and MCDM researchers are collaborating on developing such complete algorithms for multi-objective optimization [25].

Although for two or three-objective problems this may be a viable approach, for higher objective problems, a large number of solutions are required to represent the higher-dimensional Pareto-optimal front. Moreover, more sophisticated methods need to be developed for handling a large number of objectives.

- (3) **Interactive approach:** In this approach, a decision-making method is embedded within an EMO so as to find and focus near the preferred Pareto-optimal region. Some recent methodologies [6, 7] suggest some viable approaches. In the latter approach, a few clustered non-dominated trade-off solutions are shown to the decision-maker after every few generations. The decision-maker's preference information on these solutions are captured and a preference model is constructed using an utility function. The modeled utility function is then used to modify domination principles to emphasize solutions having better utility function values. The utility function is updated again with new solutions and the procedure is continued till convergence. This approach can be used for handling a large number of objectives and is a viable approach. However, the relevance of non-optimal solutions for decision-making and need for a large number of decision-maker's calls are criticisms for this approach.

10.7. Further Topics in Multi-Criterion Optimization

After the development of basic multi-objective optimization algorithms, the research in EMO have been aided with a number of other equally important developments. We discuss a few topics here.

10.7.1. *Constraint Handling*

Constraints can be simply handled by modifying the definition of domination in an EMO method.

Definition 10.3. A solution $\mathbf{x}^{(i)}$ is said to ‘constrain-dominate’ a solution $\mathbf{x}^{(j)}$ (or $\mathbf{x}^{(i)} \preceq_c \mathbf{x}^{(j)}$), if any of the following conditions are true:

- (1) Solution $\mathbf{x}^{(i)}$ is feasible and solution $\mathbf{x}^{(j)}$ is not.
- (2) Solutions $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ are both infeasible, but solution $\mathbf{x}^{(i)}$ has a smaller constraint violation.
- (3) Solutions $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ are feasible and solution $\mathbf{x}^{(i)}$ dominates solution $\mathbf{x}^{(j)}$ in the usual sense (see Definition 10.2.1).

This definition allows a feasible solution to be always dominating an infeasible solution and compared two infeasible solutions based on constraint violation values and two feasible solutions in terms of their objective values.

In the following, we show simulation results of NSGA-II applied with the above constraint handling mechanism to two test problems – the CONSTR and the problem TNK described below:

CONSTR

$$\begin{aligned}\text{Min } f_1(\mathbf{x}) &= x_1 \\ \text{Min } f_2(\mathbf{x}) &= \frac{1+x_2}{x_1} \\ x_2 + 9x_1 &\geq 6 \\ -x_2 + 9x_1 &\geq 1\end{aligned}$$

TNK

$$\begin{aligned}\text{Min } f_1(\mathbf{x}) &= x_1 \\ \text{Min } f_2(\mathbf{x}) &= x_2 \\ x_1^2 + x_2^2 - 1 - \frac{1}{10} \cos\left(16 \tan^{-1} \frac{x_1}{x_2}\right) &\geq 0 \\ (x_1 - 0.5)^2 + (x_2 - 0.5)^2 &\leq 0.5\end{aligned}$$

NSGA-II finds a good distribution of solutions on the Pareto-optimal front in both problems (Figure 10.14 and 10.15, respectively).

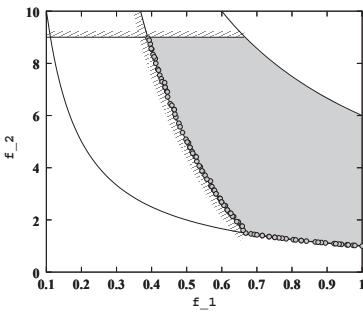


Fig. 10.14. Obtained non-dominated solutions with NSGA-II on the constrained problem CONSTR.

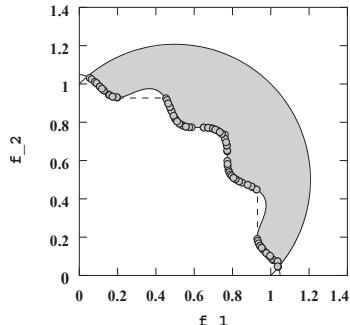


Fig. 10.15. Obtained non-dominated solutions with NSGA-II on the constrained problem TNK.

10.7.2. Many-objective Optimization

Initial studies of EMO have amply shown that EMO algorithms can be used to find a wide-spread trade-off solutions on two and three-objective optimization problems. However, their performance on four or more objective problems were not studied enough. Recently such studies have become important and are known as many-objective optimization studies in the EMO literature.

A detailed study [31] made on eight-objective problems revealed somewhat negative results about the existing EMO methodologies. But the author in his book [4] and recent other studies [32–36] have clearly explained the reasons for this behavior of EMO algorithms. EMO methodologies work by emphasizing non-dominated solutions in a population. Unfortunately, as the number of objectives increase, most population members in a randomly created population tend to become non-dominated to each other. For example, in a three-objective scenario, about 10% members in a population of size 200 are non-dominated, whereas in a 10-objective problem scenario, as high as 90% members in a population of size 200 are non-dominated. Thus, in a large-objective problem, an EMO algorithm runs out of room to introduce new population members into a generation, thereby causing a stagnation in the performance of an EMO algorithm. It has been argued that to make EMO procedures efficient, an exponentially large population size (with respect to number of objectives) is needed. This makes an EMO procedure slow and computationally less attractive.

However, practically speaking, even if an algorithm can find tens of thousands of Pareto-optimal solutions for a many-objective optimization problem, besides simply getting an idea of the nature and shape of the front, they are simply too many to be conceivable for any decision making purposes. Keeping these views in mind, EMO researchers have taken two different approaches in dealing with many-objective problems.

Recently, computationally tractable algorithms are proposed for solving 10 to 15-objective problems using pre-specified reference points [37, 38] or reference directions [39]. We discuss NSGA-III approach briefly here.

To facilitate an algorithm's ability to converge to the Pareto-optimal front and also spread solutions on the entire front uniformly, NSGA-III provides predefined and adaptive guided directions for the algorithm to help spread its solutions. NSGA-III is similar in principle and has the same feature of not having to set any new parameter. To start with, NSGA-III expects a set of supplied reference points on a M -dimensional hyperplane

making equal angle and an intercept of one at each objective axis. A set of uniformly distributed set of points are then located on the hyperplane by using any preference information or automatically without any bias by known methods, such as Das and Dennis's method [16]. These points are called reference points and are subjected to change adaptatively [38] with generations. A reference line from the origin to each of the reference points is created and kept for determining their closeness of population members for obtaining a diverse population. A typical generation of NSGA-III goes as follows. The current population P_t of size N is used to create an offspring population Q_t of size N . Both populations are combined to form a combined population R_t , which is then sorted according to different levels of non-domination. Population members are selected front-wise from first level to higher levels until all N slots of the new population P_{t+1} are filled. The last front F_l which could not be completely accepted is considered for less-crowded solutions and the all other higher-level fronts are deleted. First, population members are normalized in the objective space so that they can be compared with the reference points and reference line. On F_l , the members which lie closest to each of the reference lines are preferred. Thus, emphasis of non-dominated solutions and emphasis of closest points to each widely-distributed reference line ensures a good distributed Pareto-optimal points at the end of evolutionary process. One other feature of NSGA-III is that the population size is identical to the number of reference points and NSGA-III finds that many trade-off points at the end. Figure 10.16 shows the points obtained on three-objective DTLZ2 problem with 92 reference points. The distributing ability of NSGA-III is clear from the figure.

Figure 10.17 shows the points obtained using NSGA-III on a 10-objective DTLZ2 problem with 276 reference points. The points are plotted on a parallel coordinate plot (PCP) showing the inherent trade-off of the points. For further results and information of NSGA-III, readers are encouraged to refer to the original studies [37, 38].

10.7.3. Knowledge Extraction Through EMO

One striking difference between a single-objective optimization and multi-objective optimization is the cardinality of the solution set. In latter, multiple solutions are the outcome and each solution is theoretically an optimal solution corresponding to a particular trade-off among the objectives. Thus,

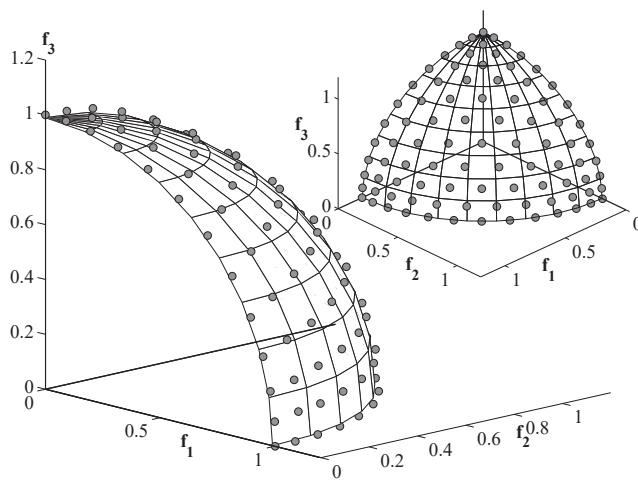


Fig. 10.16. NSGA-III solutions on the three-objective DTLZ2 problem.

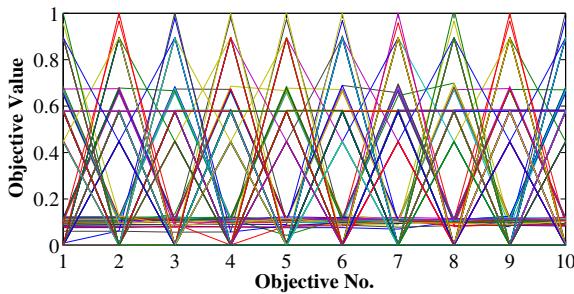


Fig. 10.17. NSGA-III solutions on the 10-objective DTLZ4 problem.

if an EMO procedure can find solutions close to the true Pareto-optimal set, what we have in our hand are a number of high-performing solutions trading-off the conflicting objectives considered in the study. Since they are all near optimal, these solutions can be analyzed for finding properties which are common to them. Such a procedure can then become a systematic approach in deciphering important and hidden properties which optimal and high-performing solutions must have for that problem. In a number of practical problem-solving tasks, the so-called *innovation* procedure is

shown to find important knowledge about high-performing solutions [40]. Such useful properties are expected to exist in practical problems [41], as they follow certain scientific and engineering principles at the core, but finding them through a systematic scientific procedure had not been paid much attention in the past. The principle of first searching for multiple trade-off and high-performing solutions using a multi-objective optimization procedure and then analyzing them to discover useful knowledge certainly remains a viable way forward. The current efforts [42, 43] to automate the knowledge extraction procedure through a sophisticated data-mining task should make the overall approach more appealing to and useful in practice.

10.8. Conclusions

The research and application in evolutionary multi-objective optimization (EMO) since early nineties has resulted in a number of efficient algorithms for finding a set of well-diversified, near Pareto-optimal solutions. EMO algorithms are now regularly being applied to different problems in most areas of science, engineering and commerce. This chapter has presented a number of classical multi-objective optimization algorithms and discussed a number of their shortcomings. Thereafter, we have discussed principles of EMO and described in detail a few popularly-used algorithms.

Three different methods of combining EMO with multi-criterion decision-making (MCDM) methods have been presented next. The methods differ from how the optimization and decision-making activities are sequenced. Based on the discussions, it has been argued that the recent interactive methods are promising and pragmatic and further studies and applications are needed to evaluate the approaches better.

Finally, a number of advanced topics have been discussed to show a general interest of the reader in the emerging EMO area. First, methods to handle non-linear constraints have been discussed. The emphasis has been made for algorithms which do not need any additional parameter. Second, recent methods to handle many objectives (10+) have been discussed. It has been clearly shown that EMO methods are now available for handling a large number of objectives, thereby allowing practitioners to study and understand interactions and trade-offs among 10 to 20 objectives. Third, EMO methods have been extended beyond simply finding a set of trade-off solutions to extract hidden principles common to multiple trade-off solutions obtained by an EMO method. The information thus obtained remains as valuable knowledge about optimal solutions of a problem

and hence the proposed ‘innovization’ (innovation through optimization) procedure should be a pragmatic strategy.

Multi-objective optimization methods are also being solved using other meta-heuristics methods, such as Particle swarm EMO [44, 45], Ant-based EMO [46, 47], and Differential evolution based EMO [48]. Simulated annealing method is used to find multiple Pareto-optimal solutions for multi-objective optimization problems [49]. Tabu search method is also used for multi-objective optimization [50]. There also exist other competent EMOs, such as strength Pareto evolutionary algorithm (SPEA) and its improved version SPEA2 [51], Pareto archived evolution strategy (PAES) and its improved versions PESA and PESA2 [52], multi-objective messy GA (MOMGA) [53], multi-objective Micro-GA [54], neighborhood constraint GA [55], ARMOGA [56] and others.

It is clear that the field of EMO research and application, in a short span, now has efficient algorithms and numerous interesting and useful applications, and has been able to attract theoretically and practically oriented researchers to come together and make collaborative activities. The practical importance of EMO’s working principle, the flexibility of evolutionary optimization which lies at the core of EMO algorithms, and demonstrated diversification of EMO’s principle to a wide variety of different problem-solving tasks are the main cornerstones for their success so far. The scope of research and application in EMO and using EMO are enormous and open-ended. This chapter remains an open invitation to everyone who is interested in any type of problem-solving tasks to take a look at what has been done in EMO and to explore how one can contribute in collaborating with EMO to address problem-solving tasks which are still in need of a better solution procedure.

Acknowledgments

Some of the contents of this chapter have been borrowed from other writings of the author on the topic.

References

- [1] V. Chankong and Y. Y. Haimes, *Multiobjective Decision Making Theory and Methodology* (New York: North-Holland, 1983).
- [2] K. Miettinen, *Nonlinear Multiobjective Optimization* (Kluwer, Boston, 1999).

- [3] R. E. Steuer, *Multiple Criteria Optimization: Theory, Computation and Application* (New York: Wiley, 1986).
- [4] K. Deb, *Multi-objective optimization using evolutionary algorithms* (Wiley, Chichester, UK, 2001).
- [5] C. A. C. Coello, D. A. VanVeldhuizen and G. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems* (Boston, MA: Kluwer, 2002).
- [6] K. Deb, A. Sinha, P. Korhonen and J. Wallenius, An interactive evolutionary multi-objective optimization method based on progressively approximated value functions, *IEEE Transactions on Evolutionary Computation* **14**(5) (2010) 723–739.
- [7] J. Branke, S. Greco, R. Slowinski and P. Zielniewicz, Interactive evolutionary multiobjective optimization using robust ordinal regression, *Proceedings of the Fifth International Conference on Evolutionary Multi-Criterion Optimization (EMO-09)*, (Berlin: Springer-Verlag, 2009), pp. 554–568.
- [8] K. Deb, S. Agrawal, A. Pratap and T. Meyarivan, A fast and elitist multi-objective genetic algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation* **6**(2) (2002) 182–197.
- [9] B. S. W. Schröder, *Ordered Sets: An Introduction* (Boston: Birkhäuser, 2003).
- [10] H. T. Kung, F. Luccio and F. P. Preparata, On finding the maxima of a set of vectors, *Journal of the Association for Computing Machinery* **22**(4) (1975) 469–476.
- [11] J. Jahn, *Vector optimization* (Berlin, Germany: Springer-Verlag, 2004).
- [12] D. H. Wolpert and W. G. Macready, No free lunch theorems for optimization, *IEEE Transactions on Evolutionary Computation* **1**(1) (1997) 67–82.
- [13] D. W. Corne and J. D. Knowles, No free lunch and free leftovers theorems for multiobjective optimisation problems, *Proceedings of the 2nd international conference on Evolutionary multi-criterion optimization, EMO'03*, (Springer-Verlag, Berlin, Heidelberg, 2003), pp. 327–341.
- [14] M. Ehrgott, *Multicriteria Optimization* (Berlin: Springer, 2000).
- [15] Y. Y. Haimes, L. S. Lasdon and D. A. Wismer, On a bicriterion formulation of the problems of integrated system identification and system optimization, *IEEE Transactions on Systems, Man, and Cybernetics* **1**(3) (1971) 296–297.
- [16] I. Das and J. Dennis, Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems, *SIAM Journal of Optimization* **8**(3) (1998) 631–657.
- [17] P. Shukla and K. Deb, On finding multiple Pareto-optimal solutions using classical and evolutionary generating methods, *European Journal of Operational Research (EJOR)* **181**(3) (2007) 1630–1652.
- [18] D. E. Goldberg, *Genetic Algorithms for Search, Optimization, and Machine Learning* (Reading, MA: Addison-Wesley, 1989).
- [19] K. Deb, L. Thiele, M. Laumanns and E. Zitzler, Scalable test problems for evolutionary multi-objective optimization, *Evolutionary Multiobjective Optimization*, eds. A. Abraham, L. Jain and R. Goldberg (London: Springer-Verlag, 2005), pp. 105–145.

- [20] K. Deb and H. Gupta, Introducing robustness in multi-objective optimization, *Evolutionary Computation Journal* **14**(4) (2006) 463–494.
- [21] K. Deb, S. Gupta, D. Daum, J. Branke, A. Mall and D. Padmanabhan, Reliability-based optimization using evolutionary algorithms, *IEEE Trans. on Evolutionary Computation* **13**(5) (2009) 1054–1074.
- [22] J. Branke, K. Deb, H. Dierolf and M. Osswald, Finding knees in multi-objective optimization, *Parallel Problem Solving from Nature (PPSN-VIII)*, (Heidelberg, Germany: Springer, 2004), pp. 722–731. LNCS 3242.
- [23] A. P. Wierzbicki, The use of reference objectives in multiobjective optimization, *Multiple Criteria Decision Making Theory and Applications*, eds. G. Fandel and T. Gal (Berlin: Springer-Verlag, 1980), pp. 468–486.
- [24] P. Korhonen and J. Laakso, A visual interactive method for solving the multiple criteria problem, *European Journal of Operational Research* **24** (1986) 277–287.
- [25] J. Branke, K. Deb, K. Miettinen and R. Slowinski, *Multiobjective optimization: Interactive and evolutionary approaches* (Springer-Verlag, Berlin, Germany, 2008).
- [26] K. Deb, J. Sundar, N. Uday and S. Chaudhuri, Reference point based multi-objective optimization using evolutionary algorithms, *International Journal of Computational Intelligence Research (IJCIR)* **2**(6) (2006) 273–286.
- [27] K. Deb and A. Kumar, Light beam search based multi-objective optimization using evolutionary algorithms, *Proceedings of the Congress on Evolutionary Computation (CEC-07)*, (2007), pp. 2125–2132.
- [28] K. Deb and A. Kumar, Interactive evolutionary multi-objective optimization and decision-making using reference direction method, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2007)*, (New York: The Association of Computing Machinery (ACM), 2007), pp. 781–788.
- [29] L. Thiele, K. Miettinen, P. Korhonen and J. Molina, A preference-based interactive evolutionary algorithm for multiobjective optimization, Tech. Rep. Working Paper Number W-412, Helsingin School of Economics, Helsingin Kauppakorkeakoulu, Finland (2007).
- [30] M. Luque, K. Miettinen, P. Eskelinen and F. Ruiz, Incorporating preference information in interactive reference point based methods for multiobjective optimization, *Omega* **37**(2) (2009) 450–462.
- [31] V. Khare, X. Yao and K. Deb, Performance scaling of multi-objective evolutionary algorithms, *Proceedings of the Second Evolutionary Multi-Criterion Optimization (EMO-03) Conference (LNCS 2632)*, (2003), pp. 376–390.
- [32] H. Ishibuchi, N. Tsukamoto and Y. Nojima, Evolutionary many-objective optimization: A short review, *Proceedings of Congress on Evolutionary Computation (CEC-2008)*, (2008), pp. 2424–2431.
- [33] J. Knowles and D. Corne, Quantifying the effects of objective space dimension in evolutionary multiobjective optimization, *Proceedings of the Fourth International Conference on Evolutionary Multi-Criterion Optimization (EMO-2007)*, (2007), pp. 757–771. (LNCS 4403).
- [34] J. A. López and C. A. C. Coello, Some techniques to deal with many-objective problems, *Proceedings of the 11th Annual Conference Companion*

- on Genetic and Evolutionary Computation Conference, (ACM, New York, 2009), pp. 2693–2696.
- [35] E. J. Hughes, Evolutionary many-objective optimisation: Many once or one many?, *IEEE Congress on Evolutionary Computation (CEC-2005)*, (2005), pp. 222–227.
 - [36] D. K. Saxena, J. A. Duro, A. Tiwari, K. Deb and Q. Zhang, Objective reduction in many-objective optimization: Linear and nonlinear algorithms, *IEEE Transactions on Evolutionary Computation* (in press).
 - [37] K. Deb and H. Jain, An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, Part I: Solving problems with box constraints, *IEEE Transactions on Evolutionary Computation* **18**(4) (2014) 577–601.
 - [38] H. Jain and K. Deb, An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, Part II: Handling constraints and extending to an adaptive approach, *IEEE Transactions on Evolutionary Computation* **18**(4) (2014) 602–622.
 - [39] Q. Zhang and H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, *Evolutionary Computation, IEEE Transactions on* **11**(6) (2007) 712–731.
 - [40] K. Deb and A. Srinivasan, Innovization: Innovating design principles through optimization., *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2006)*, New York: ACM (2006), pp. 1629–1636.
 - [41] S. Jeong, S. Hasegawa, K. Shimoyama and S. Obayashi, Development and investigation of efficient GA/PSO-hybrid algorithm applicable to real-world design optimization, *2009 IEEE Congress on Evolutionary Computation (CEC'2009)*, (Piscatway, NJ: IEEE Press, 2009), pp. 777–784.
 - [42] S. Bandaru and K. Deb, Towards automating the discovery of certain innovative design principles through a clustering based optimization technique, *Engineering optimization* **43**(9) (2011) 911–941.
 - [43] S. Bandaru and K. Deb, Automated innovization for simultaneous discovery of multiple rules in bi-objective problems, *Proceedings of Sixth International Conference on Evolutionary Multi-Criterion Optimization (EMO-2011)*, (Heidelberg: Springer, 2011), pp. 1–15.
 - [44] C. A. C. Coello and M. S. Lechuga, MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization, *Congress on Evolutionary Computation (CEC'2002)*, **2**, (IEEE Service Center, Piscataway, New Jersey, May 2002), pp. 1051–1056.
 - [45] S. Mostaghim and J. Teich, Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO), *2003 IEEE Swarm Intelligence Symposium Proceedings*, (IEEE Service Center, Indianapolis, Indiana, USA, April 2003), pp. 26–33.
 - [46] P. R. McMullen, An ant colony optimization approach to addressing a JIT sequencing problem with multiple objectives, *Artificial Intelligence in Engineering* **15** (2001) 309–317.
 - [47] M. Gravel, W. L. Price and C. Gagné, Scheduling continuous casting of aluminum using a multiple objective ant colony optimization metaheuristic,

- European Journal of Operational Research* **143** (November 2002) 218–229.
- [48] B. Babu and M. L. Jehan, Differential Evolution for Multi-Objective Optimization, *Proceedings of the 2003 Congress on Evolutionary Computation (CEC'2003)*, **4**, (IEEE Press, Canberra, Australia, December 2003), pp. 2696–2703.
 - [49] S. Bandyopadhyay, S. Saha, U. Maulik and K. Deb, A simulated annealing-based multiobjective optimization algorithm: Amosa, *IEEE Trans. Evolutionary Computation* **12**(3) (2008) 269–283.
 - [50] M. P. Hansen, Tabu search in multiobjective optimization: MOTS (1997), Paper presented at The Thirteenth International Conference on Multi-Criterion Decision Making (MCDM'97), University of Cape Town.
 - [51] E. Zitzler, M. Laumanns and L. Thiele, SPEA2: Improving the strength Pareto evolutionary algorithm for multiobjective optimization, *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, ed. K. C. G. et al. (International Center for Numerical Methods in Engineering (CIMNE), 2001), pp. 95–100.
 - [52] D. W. Corne, J. D. Knowles and M. Oates, The Pareto envelope-based selection algorithm for multiobjective optimization, *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature VI (PPSN-VI)*, (2000), pp. 839–848.
 - [53] D. V. Veldhuizen and G. B. Lamont, Multiobjective evolutionary algorithms: Analyzing the state-of-the-art, *Evolutionary Computation Journal* **8**(2) (2000) 125–148.
 - [54] C. A. C. Coello and G. Toscano, A micro-genetic algorithm for multi-objective optimization, Tech. Rep. Lania-RI-2000-06, Laboratoria Nacional de Informatica AvRyerkerk, M., Averill, R., Deb, K., and Goodman, E. (2012). Optimization for Variable-Size Problems Using Genetic Algorithms. *Proceedings of the 14th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. (Indianapolis, USA). AIAA 2012-5569, Reston, VA: AIAA.anzada, Xalapa, Veracruz, Mexico (2000).
 - [55] D. H. Loughlin and S. Ranjithan, The neighborhood constraint method: A multiobjective optimization technique, *Proceedings of the Seventh International Conference on Genetic Algorithms*, (1997), pp. 666–673.
 - [56] D. Sasaki, M. Morikawa, S. Obayashi and K. Nakahashi, Aerodynamic shape optimization of supersonic wings by adaptive range multiobjective genetic algorithms, *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO 2001)*, (2001), pp. 639–652.

Chapter 11

Rough Sets in Pattern Recognition

Andrzej Skowron¹, Hung Son Nguyen² and Andrzej Jankowski³

¹*Institute of Mathematics
Warsaw University, Warsaw, Poland
and
Systems Research Institute
Polish Academy of Sciences, Warsaw, Poland
skowron@mimuw.edu.pl*

²*Institute of Mathematics
Warsaw University, Warsaw, Poland
son@mmuw.edu.pl*

³*The Dziubanski Foundation of Knowledge Technology
Nowogrodzka, Warsaw, Poland
Andrzej.AdgaM@gmail.com*

To Roman Swiniarski in Memoriam

In this chapter we begin from rough set rudiments. Next we outline applications of rough sets in pattern recognition. This chapter covers partially the material presented in the chapter [272] published in 2001 in the book [184] and also includes information on development of the rough set based methods in pattern recognition published after 2001.

11.1. Introduction

The rough set (RS) approach was proposed by Professor Zdzisław Pawlak in 1982 [203, 204] as a tool for dealing with imperfect knowledge, in particular with vague concepts. Over the years many applications of methods based on RS theory alone or in combination with other approaches have been developed.

The RS philosophy is founded on the assumption that with every object of the universe of discourse we associate some information (data, knowledge). Objects characterized by the same information are indiscernible (similar) in view of the available information about them. The *indiscernibility relation* generated in this way is the mathematical basis of RS theory. This understanding of indiscernibility is related to the idea of Gottfried Wilhelm Leibniz that objects are indiscernible if and only if all available functionals take on them identical values (Leibniz's Law of Indiscernibility: The Identity of Indiscernibles) [112]. However, in the RS approach indiscernibility is defined relative to a given set of functionals (attributes).

Any set of all indiscernible (similar) objects is called an elementary set, and forms a basic granule (atom) of knowledge about the universe. Any union of some elementary sets is referred to as *crisp* (precise) set. If a set is not crisp then it is called *rough* (imprecise, vague). Note that due to the computational complexity of searching for relevant crisp sets for the considered problem, the searching is usually restricted to a feasible subfamily of the family of all possible unions of elementary sets.

Consequently, each RS has *borderline cases*, i.e., objects which cannot be classified with certainty as members of either the set or its complement. Obviously crisp sets have no borderline elements at all. This means that borderline cases cannot be properly classified by employing available knowledge.

Thus, the assumption that objects can be “seen” only through the information available about them leads to the view that knowledge has granular structure. Due to the granularity of knowledge, some objects of interest cannot be discerned and appear as the same (or similar). As a consequence, vague concepts in contrast to precise concepts, cannot be characterized in terms of information about their elements. Therefore, in the proposed approach, we assume that any vague concept is replaced by a pair of precise concepts – called the lower and the upper approximation of the vague concept. The lower approximation consists of all objects which definitely belong to the concept and the upper approximation contains all objects which possibly belong to the concept. The difference between the upper and the lower approximation constitutes the boundary region of the vague concept. Approximations are two basic operations in RS theory. Hence, RS theory expresses vagueness not by means of membership, but by employing a boundary region of a set. If the boundary region of a set is empty it means that the set is crisp, otherwise the set is rough (inexact). A nonempty boundary region of a set means that our knowledge about the set is not sufficient to define the set precisely.

In this chapter, we begin from RS rudiments. We emphasize applications of rough sets in pattern recognition based on (approximate) boolean reasoning. This includes in particular applications of rough sets in feature (attribute) selection, constructive induction of new features (in particular, by discretization or symbolic value grouping), clustering, inducing classifiers, ontology approximation, discovery of behavioral patterns and models of processes from data.

This chapter covers the material presented in the chapter [272] published in 2001 in the book [184] and also includes information on development of the RS based methods in pattern recognition published after 2001 (see, e.g., [169]). For further readings the reader is also referred to [2, 74, 80, 90, 98, 116, 133, 134, 147, 148, 185, 186, 191, 193, 208, 214, 240, 254, 256, 257, 288, 293, 328, 329]. In this chapter, due to the space limitation, we do not discuss the RS based clustering methods (see, e.g., [96, 125, 126, 146, 149–151, 212]).

In the development of RS theory and applications, one can distinguish three main stages. While the first period was based on the assumption that objects are perceived by means of partial information represented by attributes, in the second period it was assumed that information about the approximated concepts is partial too. Approximation spaces and searching strategies for relevant approximation spaces were recognized as the basic tools for rough sets. Important achievements both in theory and applications were obtained. Nowadays, a new period for rough sets is emerging which is also briefly characterized in the chapter.

The RS approach seems to be of fundamental importance in artificial intelligence and cognitive sciences, especially in machine learning, data mining and knowledge discovery from databases, pattern recognition, decision support systems, expert systems, intelligent systems, multiagent systems, adaptive systems, autonomous systems, inductive reasoning, commonsense reasoning, adaptive judgement, conflict analysis.

Rough sets have established relationships with many other approaches such as fuzzy set theory, granular computing, evidence theory, formal concept analysis, (approximate) boolean reasoning, multicriteria decision analysis, statistical methods, decision theory, matroids, logical data analysis, test theory have been clarified. Despite the overlap with many other theories RS theory may be considered as an independent discipline in its own right. There are reports on many hybrid methods obtained by combining rough sets with other approaches such as soft computing (fuzzy sets, neural networks, genetic algorithms), statistics, natural computing, mereo-

ogy, principal component analysis, singular value decomposition or support vector machines.

The main advantage of RS theory in data analysis is that it does not necessarily need any preliminary or additional information about data like probability distributions in statistics, basic probability assignments in evidence theory, a grade of membership or the value of possibility in fuzzy set theory.

The basic ideas of RS theory and its extensions as well as many interesting applications can be found in many books, issues of the Transactions on Rough Sets journal, special issues of other journals, numerous proceedings of international conferences, and tutorials. For further details the reader is referred to the basic papers by Professor Zdzisław Pawlak [203, 204], survey papers or tutorials (see, *e.g.*, [30, 208–210, 271]), references included in the mentioned articles and books as well as to the web pages www.roughsets.org, rsds.univ.rzeszow.pl.

In this chapter, we begin with a short discussion on vague concepts (see Sect. 11.2). Next, we recall the basic concepts of RS theory (see Sect. 11.3). Some extensions of the RS approach are outlined in Sect. 11.4. In Sect. 11.5, we discuss the relationship of the RS approach with inductive reasoning. In particular, we present the RS approach to inducing RS based classifiers and inducing relevant approximation spaces. The RS approach based on combination of rough sets and boolean reasoning with applications in pattern recognition, machine learning, and data mining is outlined in Sect. 11.6. An illustrative example related to scalable RS methods is presented in Sect. 11.7. The results of combination of the RS approach with other approaches such as fuzzy sets or neural networks are included in Sect. 11.8. Next, we discuss some challenging issues for rough sets (see Sect. 11.9). We start from an extension of Granular Computing (GrC) to Interactive (Rough) Granular Computing (I(R)GrC) for dealing with computations on complex granules progressing due to interactions of physical objects. Methods based on I(R)GrC are proposed to deal with such issues as approximation of complex vague concepts, ontology approximation, context inducing, process mining, perception based computing (PBC). Moreover, we propose to build foundations for the WisTech program [92, 93] on the basis of I(R)GrC. Some comments on relationships of rough sets and logic from the point of view of pattern discovery are included in Sect. 11.10. Sect. 11.11 concludes the chapter.

11.2. Vague Concepts

Mathematics requires that all mathematical notions (including set) must be exact, otherwise precise reasoning would be impossible. However, philosophers [101] and recently computer scientists as well as other researchers have become interested in *vague* (imprecise) concepts. Moreover, in the XX century one can observe the drift paradigms in modern science from dealing with precise concepts to vague concepts, especially in the case of complex systems (*e.g.*, in economy, biology, psychology, sociology, quantum mechanics).

In classical set theory, a set is uniquely determined by its elements. In other words, this means that every element must be uniquely classified as belonging to the set or not. That is to say the notion of a set is a *crisp* (precise) one. For example, the set of odd numbers is crisp because every integer is either odd or even. In contrast to odd numbers, the notion of a beautiful painting is vague, because we are unable to classify uniquely all paintings into two classes: *beautiful* and not *beautiful*. Some paintings cannot be decided whether they are beautiful or not and thus they remain in the doubtful area. Thus, *beauty* is not a precise but a vague concept.

Almost all concepts we are using in natural language are vague. Therefore, common sense reasoning based on natural language must be based on vague concepts and not on classical logic. Interesting discussion of this issue can be found in [235]. The idea of vagueness can be traced back to the ancient Greek philosopher Eubulides of Megara (ca. 400BC) who first formulated so called “sorites” (heap) and “falakros” (bald man) paradoxes [101]. There is a huge literature on issues related to vagueness and vague concepts in philosophy (see, *e.g.*, references in the book [101]).

Vagueness is often associated with the boundary region approach (*i.e.*, existence of objects which cannot be uniquely classified relative to a set or its complement) which was first formulated in 1893 by the father of modern logic, German logician, Gottlob Frege (1848–1925) [45]. According to Frege (see *Grundgesetze der Arithmetik*, vol. ii, Sect. 56 [45, 53]) the concept must have a sharp boundary. *To the concept without a sharp boundary there would correspond an area that would not have any sharp boundary – line all around.* It means that mathematics must use crisp, not vague concepts, otherwise it would be impossible to reason precisely. However, vagueness in opinion of Ludwig Wittgenstein is an essential feature of language. A language is not a calculus with rigid rules that provide for all possible circumstances. There are many vague concepts in natural

language [4, 28]. One should also note that vagueness also relates to insufficient specificity, as the result of lack of feasible searching methods for sets of features adequately describing concepts.

Discussion on vague (imprecise) concepts in philosophy includes the following characteristic features of them [101]: (i) the presence of borderline cases, (ii) boundary regions of vague concepts are not crisp, (iii) vague concepts are susceptible to sorites paradoxes. In the sequel we discuss the first two issues in the RS framework. The reader can find in the book [235] the discussion on application of the RS approach to vagueness.

11.3. Rudiments of Rough Sets

This section briefly delineates basic concepts in RS theory.

11.3.1. Indiscernibility and Approximation

The starting point of RS theory is the indiscernibility relation, which is generated by information about objects of interest (see Sect. 11.1). The indiscernibility relation expresses the fact that due to a lack of information (or knowledge) we are unable to discern some objects employing available information (or knowledge). This means that, in general, we are unable to deal with each particular object but we have to consider granules of indiscernible objects as a fundamental basis for our theory.

From a practical point of view, it is better to define basic concepts of this theory in terms of data. Therefore we will start our considerations from a data set called an *information system*. An information system is represented by a data table containing rows labeled by objects of interest, columns labeled by attributes and entries of the table are attribute values. For example, a data table can describe a set of patients in a hospital. The patients can be characterized by some attributes, like *age*, *sex*, *blood pressure*, *body temperature*, etc. With every attribute a set of its values is associated, e.g., values of the attribute *age* can be *young*, *middle*, and *old*. Attribute values can be also numerical. In data analysis the basic problem we are interested in is to find *patterns* in data, i.e., to find a relationship between some sets of attributes, e.g., we might be interested whether *blood pressure* depends on *age* and *sex*.

Suppose we are given a pair $\mathbb{A} = (U, A)$ of non-empty, finite sets U and A , where U is the *universe of objects*, and A – a set consisting of *attributes*, i.e. functions $a : U \rightarrow V_a$, where V_a is the set of values of attribute

a , called the *domain* of a . The pair $\mathbb{A} = (U, A)$ is called an *information system* [202]. Any information system can be represented by a data table with rows labeled by objects and columns labeled by attributes^a. Any pair (x, a) , where $x \in U$ and $a \in A$ defines the table entry consisting of the value $a(x)$.

Any subset B of A determines a binary relation $I\mathcal{N}\mathcal{D}_B$ on U , called an *indiscernibility relation*, defined by

$$x I\mathcal{N}\mathcal{D}_B y \text{ if and only if } a(x) = a(y) \text{ for every } a \in B, \quad (11.1)$$

where $a(x)$ denotes the value of attribute a for object x .

Obviously, $I\mathcal{N}\mathcal{D}_B$ is an equivalence relation. The family of all equivalence classes of $I\mathcal{N}\mathcal{D}_B$, i.e., the partition determined by B , will be denoted by $U/I\mathcal{N}\mathcal{D}_B$, or simply U/B ; an equivalence class of $I\mathcal{N}\mathcal{D}_B$, i.e., the block of the partition U/B , containing x will be denoted by $B(x)$ (other notation used: $[x]_B$ or more precisely $[x]_{I\mathcal{N}\mathcal{D}_B}$). Thus in view of the data we are unable, in general, to observe individual objects but we are forced to reason only about the accessible granules of knowledge [194, 204, 229].

If $(x, y) \in I\mathcal{N}\mathcal{D}_B$ we will say that x and y are *B-indiscernible*. Equivalence classes of the relation $I\mathcal{N}\mathcal{D}_B$ (or blocks of the partition U/B) are referred to as *B-elementary sets* or *B-elementary granules*. In the RS approach the elementary sets are the basic building blocks (concepts) of our knowledge about reality. The unions of *B-elementary sets* are called *B-definable sets*.

For $B \subseteq A$ we denote by $Inf_B(x)$ the *B-signature* of $x \in U$, i.e., the set $\{(a, a(s)) : a \in B\}$. Let $INF(B) = \{Inf_B(s) : s \in U\}$. Then for any objects $x, y \in U$ the following equivalence holds: $x I\mathcal{N}\mathcal{D}_By$ if and only if $Inf_B(x) = Inf_B(y)$.

The indiscernibility relation will be further used to define basic concepts of RS theory. Let us define now the following two operations on sets $X \subseteq U$

$$\text{LOW}_B(X) = \{x \in U : B(x) \subseteq X\}, \quad (11.2)$$

$$\text{UPP}_B(X) = \{x \in U : B(x) \cap X \neq \emptyset\}, \quad (11.3)$$

assigning to every subset X of the universe U two sets $\text{LOW}_B(X)$ and $\text{UPP}_B(X)$ called the *B-lower* and the *B-upper approximation* of X , respectively. The set

$$\text{BN}_B(X) = \text{UPP}_B(X) - \text{LOW}_B(X), \quad (11.4)$$

^aNote, that in statistics or machine learning such a data table is called a sample [81]. One can also compare data tables corresponding to information systems with relations in relational databases [52].

will be referred to as the *B-boundary region* of X .

From the definition we obtain the following interpretation:

- The *lower approximation* of a set X with respect to B is the set of all objects, which can be for *certain* classified as objects in X using B (are *certainly* in X in view of B).
- The *upper approximation* of a set X with respect to B is the set of all objects which can be *possibly* classified as objects in X using B (are *possibly* in X in view of B).
- The *boundary region* of a set X with respect to B is the set of all objects, which can be classified neither as in X nor as in $U - X$ using B .

In other words, due to the granularity of knowledge, rough sets cannot be characterized by using available knowledge. Therefore with every RS we associate two *crisp* sets, called *lower* and *upper approximation*. Intuitively, the lower approximation of a set consists of all elements that *surely* belong to the set, whereas the upper approximation of the set constitutes of all elements that *possibly* belong to the set, and the *boundary region* of the set consists of all elements that cannot be classified uniquely to the set or its complement, by employing available knowledge. The approximation definition is clearly depicted in Figure 11.1.

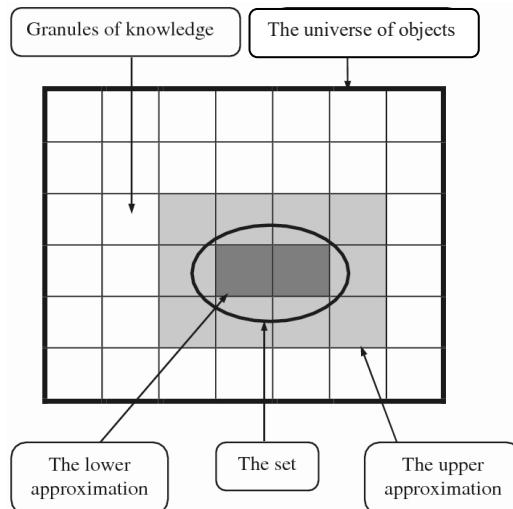


Fig. 11.1. A rough set.

Now we are ready to give the definition of rough sets.

If the boundary region of X is the empty set, i.e., $\text{BN}_B(X) = \emptyset$, then the set X is *crisp (exact)* with respect to B ; in the opposite case, i.e., if $\text{BN}_B(X) \neq \emptyset$, the set X is referred to as *rough (inexact)* with respect to B . Thus any RS, in contrast to a *crisp set*, has a non-empty boundary region.

Thus a set is *rough (imprecise)* if it has nonempty boundary region; otherwise the set is crisp (precise). This is exactly the idea of vagueness proposed by Frege.

Let us observe that the definition of rough sets refers to data (knowledge), and is *subjective*, in contrast to the definition of classical sets, which is in some sense an *objective* one.

A RS can also be characterized numerically by the following coefficient

$$\alpha_B(X) = \frac{|\text{LOW}_B(X)|}{|\text{UPP}_B(X)|}, \quad (11.5)$$

called the *accuracy of approximation*, where X is a nonempty set and $|S|$ denotes the cardinality of set S .^b Obviously $0 \leq \alpha_B(X) \leq 1$. If $\alpha_B(X) = 1$ then X is *crisp* with respect to B (X is *precise* with respect to B), and otherwise, if $\alpha_B(X) < 1$ then X is *rough* with respect to B (X is *vague* with respect to B). The accuracy of approximation (*roughness*) can be used to measure the quality of approximation of decision classes on the universe U . One can use another measure of accuracy defined by $1 - \alpha_B(X)$ or by $1 - \frac{|\text{BN}_B(X)|}{|U|}$. Some other measures of approximation accuracy are also used, e.g., based on entropy or some more specific properties of boundary regions [54, 253, 284]. The choice of a relevant accuracy of approximation depends on a particular data set. Observe that the accuracy of approximation of X can be tuned by B . Another approach to accuracy of approximation can be based on the Variable Precision Rough Set Model (VPRSM) [343].

The proposed in the paper [195] rough entropy measure may be used as yet another measure of roughness. This measure was successfully applied in image analysis [51, 135, 195, 243].

In the book by Keefe [101], it is stressed that boundaries of vague concepts are not crisp. In the definition presented in this chapter, the notion of boundary region is defined as a crisp set $\text{BN}_B(X)$. However, let us observe that this definition is relative to the subjective knowledge expressed by attributes from B . Different sources of information may use different sets of attributes for concept approximation. Hence, the boundary region can

^bthe cardinality of set S is also denoted by “ $\text{card}(S)$ ” instead of $|S|$.

change when we consider these different views. Another reason for boundary change may be related to incomplete information about concepts. They are known only on samples of objects [81]. Hence, when new objects appear again the boundary region may change. From the discussion in the literature it follows that vague concepts cannot be approximated with satisfactory quality by *static* constructs such as induced membership inclusion functions, approximations or models derived, *e.g.*, from a sample. Understanding of vague concepts can be only realized in a process in which the induced models are adaptively matching the concepts in dynamically changing environments. This conclusion seems to have important consequences for further development of RS theory in combination with fuzzy sets and other soft computing paradigms for adaptive approximate reasoning. For further details the reader is referred, *e.g.*, to articles [251, 286, 339].

In the next section, we discuss decision rules (constructed over a selected set B of features or a family of sets of features) which are used in inducing classification algorithms (classifiers) making it possible to classify to decision classes unseen objects (see Sect. 11.5). Parameters which are tuned in searching for a classifier with the high quality are its description size (defined using decision rules) and its quality of classification (measured by the number of misclassified objects on a given set of objects). By selecting a proper balance between the accuracy of classification and the description size we expect to find the classifier with the high quality of classification also on unseen objects. This approach is based on the minimum description length principle [236, 237, 285].

The reader is referred to the following references [8, 15, 163, 169] for further readings on the RS based classification methods. In the literature are also discussed more advances issues such as (i) applications of rough sets in searching for outliers [117], (ii) RS methods for imbalanced data [127]), or (iii) rough sets in incremental learning [340].

11.3.2. Decision Systems and Decision Rules

Sometimes we distinguish in an information system $\mathbb{A} = (U, A)$ a partition of A into two disjoint classes $C, D \subseteq A$ of attributes, called *condition* and *decision (action)* attributes, respectively. The tuple $\mathbb{A} = (U, C, D)$ is called a *decision system* (or decision table).

Let $V = \bigcup\{V_a \mid a \in C\} \bigcup \{V_d \mid d \in D\}$. Atomic formulae over $B \subseteq C \cup D$ and V are expressions $a = v$ called *descriptors (selectors) over B and V*, where $a \in B$ and $v \in V_a$. The set of formulae over B and V , denoted by

$\mathcal{F}(B, V)$, is the least set containing all atomic formulae over B and V and closed under the propositional connectives \wedge (conjunction), \vee (disjunction) and \neg (negation).

By $\|\varphi\|_{\mathbb{A}}$ we denote the meaning of $\varphi \in \mathcal{F}(B, V)$ in the decision table \mathbb{A} which is the set of all objects in U with the property φ . These sets are defined by $\|a = v\|_{\mathbb{A}} = \{x \in U \mid a(x) = v\}$, $\|\varphi \wedge \varphi'\|_{\mathbb{A}} = \|\varphi\|_{\mathbb{A}} \cap \|\varphi'\|_{\mathbb{A}}$; $\|\varphi \vee \varphi'\|_{\mathbb{A}} = \|\varphi\|_{\mathbb{A}} \cup \|\varphi'\|_{\mathbb{A}}$; $\|\neg\varphi\|_{\mathbb{A}} = U - \|\varphi\|_{\mathbb{A}}$. The formulae from $\mathcal{F}(C, V)$, $\mathcal{F}(D, V)$ are called *condition formulae of \mathbb{A}* and *decision formulae of \mathbb{A}* , respectively.

Any object $x \in U$ belongs to the *decision class* $\|\bigwedge_{d \in D} d = d(x)\|_{\mathbb{A}}$ of \mathbb{A} . All decision classes of \mathbb{A} create a partition U/D of the universe U .

A *decision rule* for \mathbb{A} is any expression of the form $\varphi \Rightarrow \psi$, where $\varphi \in \mathcal{F}(C, V)$, $\psi \in \mathcal{F}(D, V)$, and $\|\varphi\|_{\mathbb{A}} \neq \emptyset$. Formulae φ and ψ are referred to as the *predecessor* and the *successor* of decision rule $\varphi \Rightarrow \psi$. Decision rules are often called “*IF ... THEN ...*” rules. Such rules are used in machine learning [81].

Decision rule $\varphi \Rightarrow \psi$ is *true* in \mathbb{A} if and only if $\|\varphi\|_{\mathbb{A}} \subseteq \|\psi\|_{\mathbb{A}}$. Otherwise, one can measure its *truth degree* by introducing some inclusion measure of $\|\varphi\|_{\mathbb{A}}$ in $\|\psi\|_{\mathbb{A}}$. Let us denote by $|\varphi|$ the number of objects from U that satisfies formula φ , *i.e.*, the cardinality of $\|\varphi\|_{\mathbb{A}}$. According to Lukasiewicz

[128], one can assign to formula φ the value $\frac{|\varphi|}{|U|}$, and to the implication

$\varphi \Rightarrow \psi$ the fractional value $\frac{|\varphi \wedge \psi|}{|\varphi|}$, under the assumption that $\|\varphi\| \neq \emptyset$. Proposed by Lukasiewicz, that fractional part was much later adapted by the machine learning and data mining communities, *e.g.*, in the definitions of the accuracy of decision rules or confidence of association rules.

Each object x of a decision system determines a *decision rule*

$$\bigwedge_{a \in C} a = a(x) \Rightarrow \bigwedge_{d \in D} d = d(x). \quad (11.6)$$

For any decision table $\mathbb{A} = (U, C, D)$ one can consider a *generalized decision* function $\partial_A : U \longrightarrow \mathcal{P}(\text{INF}(D))$ defined by

$$\partial_A(x) = \left\{ i \in \text{INF}(D) : \exists x' \in U [(x', x) \in \text{IN}\mathcal{D}_C \text{ and } \text{Inf}_D(x') = i] \right\}, \quad (11.7)$$

where $A = C \cup D$, $\mathcal{P}(\text{INF}(D))$ is the powerset of the set $\text{INF}(D)$ of all possible decision signatures.

The decision table \mathbb{A} is called *consistent (deterministic)*, if $|\partial_A(x)| = 1$, for any $x \in U$. Otherwise \mathbb{A} is said to be *inconsistent (non-deterministic)*.

Hence, a decision table is inconsistent if it consists of some objects with different decisions but indiscernible with respect to condition attributes. Any set consisting of all objects with the same generalized decision value is called a *generalized decision class*.

Now, one can consider certain (possible) rules [67, 71] for decision classes defined by the lower (upper) approximations of such generalized decision classes of \mathbb{A} . This approach can be extend, using the relationships of rough sets with the Dempster-Shafer theory [244, 253], by considering rules relative to decision classes defined by the lower approximations of unions of decision classes of \mathbb{A} .

Numerous methods have been developed for generation of different types of decision rules, and the reader can find them in the literature on rough sets. Usually, one is searching for decision rules (semi) optimal with respect to some optimization criteria describing quality of decision rules in concept approximations.

11.3.3. Dependency of Attributes

Another important issue in data analysis is discovering dependencies between attributes in a given decision system $\mathbb{A} = (U, C, D)$. Intuitively, a set of attributes D depends totally on a set of attributes C , denoted $C \Rightarrow D$, if the values of attributes from C uniquely determine the values of attributes from D . In other words, D depends totally on C , if there exists a functional dependency between values of C and D . Hence, $C \Rightarrow D$ if and only if the rule (11.6) is true on \mathbb{A} for any $x \in U$. D can depend partially on C . Formally such a dependency can be defined in the following way.

We will say that D depends on C to a degree k ($0 \leq k \leq 1$), denoted $C \Rightarrow_k D$, if

$$k = \gamma(C, D) = \frac{|POS_C(D)|}{|U|}, \quad (11.8)$$

where

$$POS_C(D) = \bigcup_{X \in U/D} \text{LOW}_C(X), \quad (11.9)$$

called a *positive region* of the partition U/D with respect to C , is the set of all elements of U that can be uniquely classified to blocks of the partition U/D , by means of C .

If $k = 1$ we say that D depends totally on C , and if $k < 1$, we say that D depends partially (to degree k) on C . If $k = 0$ then the positive region of the partition U/D with respect to C is empty.

The coefficient k expresses the ratio of all elements of the universe, which can be properly classified to blocks of the partition U/D , employing attributes C and will be called the *degree of the dependency*.

It can be easily seen that if D depends totally on C then $I\mathcal{N}\mathcal{D}_C \subseteq I\mathcal{N}\mathcal{D}_D$. It means that the partition generated by C is finer than the partition generated by D . Notice, that the concept of dependency discussed above corresponds to that considered in relational databases.

Observe, that (11.8) defines only one of possible measures of dependency between attributes [283]. One also can compare the dependency discussed in this section with dependencies considered in databases [52].

11.3.4. Reduction of Attributes

We often face a question whether we can remove some data from a data-table preserving its basic properties, that is – whether a table contains some superfluous data.

Let us express this idea more precisely.

Let $C, D \subseteq A$, be sets of condition and decision attributes respectively. We will say that $C' \subseteq C$ is a *D-reduct* (reduct with respect to D) of C , if C' is a minimal subset of C such that

$$\gamma(C, D) = \gamma(C', D). \quad (11.10)$$

The intersection of all D -reducts is called a *D-core* (core with respect to D). Because the core is the intersection of all reducts, it is included in every reduct, i.e., each element of the core belongs to some reduct. Thus, in a sense, the core is the most important subset of attributes, since none of its elements can be removed without affecting the classification power of attributes. Certainly, the geometry of reducts can be more compound. For example, the core can be empty but there can exist a partition of reducts into a few sets with non empty intersection.

Many other kinds of reducts and their approximations are discussed in the literature [18, 35, 163, 166, 170, 245, 278, 280, 284, 285, 290]. For example, if one change the condition (11.10) to $\partial_A(x) = \partial_B(x)$, (where $A = C \cup D$ and $B = C' \cup D$) then the defined reducts are preserving the generalized decision. Other kinds of reducts are preserving, e.g.: (i) the distance between attribute value vectors for any two objects, if this distance is greater than a given threshold [245], (ii) the distance between entropy distributions between any two objects, if this distance exceeds a given threshold [278, 284], or (iii) the so called reducts relative to object

used for generation of decision rules [18]. There are some relationships between different kinds of reducts. If B is a reduct preserving the generalized decision, than in B is included a reduct preserving the positive region. For mentioned above reducts based on distances and thresholds one can find analogous dependency between reducts relative to different thresholds. By choosing different kinds of reducts we select different degrees to which information encoded in data is preserved. Reducts are used for building data models. Choosing a particular reduct or a set of reducts has impact on the model size as well as on its quality in describing a given data set. It is worthwhile mentioning that randomly generated reduct can't be relevant for classifier inducing. The model size together with the model quality are two basic components tuned in selecting relevant data models. This is known as the *minimum description length principle* [236, 237, 284, 285]. Selection of relevant kinds of reducts is an important step in building data models. It turns out that the different kinds of reducts can be efficiently computed using heuristics based, *e.g.*, on the boolean reasoning approach [27].

11.3.5. Rough Membership

Let us observe that rough sets can be also defined employing the rough membership function (see Eq. 11.11) instead of approximation [207]. That is, consider

$$\mu_X^B : U \rightarrow [0, 1],$$

defined by

$$\mu_X^B(x) = \frac{|B(x) \cap X|}{|B(X)|}, \quad (11.11)$$

where $x \in X \subseteq U$. The value $\mu_X^B(x)$ can be interpreted as the degree that x belongs to X in view of knowledge about x expressed by B or the degree to which the elementary granule $B(x)$ is included in the set X . This means that the definition reflects a subjective knowledge about elements of the universe, in contrast to the classical definition of a set.

The rough membership function can also be interpreted as the conditional probability that x belongs to X given B . This interpretation was used by several researchers in the RS community [69, 284, 314, 322, 333, 343, 344]. Note also that the ratio on the right hand side of the Equation (11.11) is known as the confidence coefficient in data mining [81, 104]. It is worthwhile to mention that set inclusion to a degree has been considered by Lukasiewicz [128] in studies on assigning fractional truth values to logical formulas.

Some other measures describing the relationships between $B(x)$ and X are also used such as covering [312].

One can observe that the rough membership function has the following properties [207]:

- 1) $\mu_X^B(x) = 1$ iff $x \in \text{LOW}_B(X)$,
- 2) $\mu_X^B(x) = 0$ iff $x \in U - \text{UPP}_B(X)$,
- 3) $0 < \mu_X^B(x) < 1$ iff $x \in \text{BN}_B(X)$,
- 4) $\mu_{U-X}^B(x) = 1 - \mu_X^B(x)$ for any $x \in U$,
- 5) $\mu_{X \cup Y}^B(x) \geq \max(\mu_X^B(x), \mu_Y^B(x))$ for any $x \in U$,
- 6) $\mu_{X \cap Y}^B(x) \leq \min(\mu_X^B(x), \mu_Y^B(x))$ for any $x \in U$.

From the properties it follows that the rough membership differs essentially from the fuzzy membership [336], for properties 5) and 6) show that the membership for union and intersection of sets, in general, cannot be computed – as in the case of fuzzy sets – from their constituents membership. Thus formally the rough membership is different from fuzzy membership. Moreover, the rough membership function depends on an available knowledge (represented by attributes from B). Besides, the rough membership function, in contrast to fuzzy membership function, has a probabilistic flavor.

One of the consequences of perceiving objects by information about them is that for some objects one cannot decide if they belong to a given set or not. However, one can estimate the degree to which objects belong to sets. This is a crucial observation in building foundations for approximate reasoning. Dealing with imperfect knowledge implies that one can only characterize satisfiability of relations between objects to a degree, not precisely. One of the fundamental relations on objects is a rough inclusion relation describing that objects are parts of other objects to a degree. The rough mereological approach [194, 222–224, 226] based on such a relation is an extension of the Leśniewski mereology [113].

11.4. Generalizations of Approximation Spaces

The original approach by Professor Pawlak was based on indiscernibility defined by equivalence relations. Any such indiscernibility relation defines a partition of the universe of objects. Over the years many generalizations of this approach were introduced many of which are based on coverings rather than partitions. In particular, one can consider similarity (tolerance) based RS approach, binary relation based rough sets, neighborhood

and covering rough sets, dominance based RS approach, hybridization of rough sets and fuzzy sets, probabilistic RS approach, rough sets on abstract algebraic structures, and several others (see, e.g., [90, 288, 328, 329]).

One should note that dealing with coverings requires solving several new algorithmic problems such as selection of family of definable sets or resolving problems with selection of relevant definition of approximation of sets among many possible ones. Moreover, for a given problem (e.g., classification problem) one should discover the relevant covering for the target classification task. In the literature there are numerous papers dedicated to theoretical aspects of the covering RS approach (see, e.g., [328]). However, still much more work should be done on rather hard algorithmic issues for the relevant covering discovery.

Another issue to be solved is related to inclusion measures. Parameters of such measures are tuned to induce of the high quality approximations. Usually, this is done on the basis of the minimum description length principle. In particular, approximation spaces with rough inclusion measures have been investigated. This approach was further extended to rough mereological approach. More general cases of approximation spaces with rough inclusion were also discussed in the literature including approximation spaces in Granular Computing (GrC). Finally, it is worthwhile to mention the approach for ontology approximation used in hierarchical learning of complex vague concepts.

In this section we discuss some aspects of the above mentioned issues.

Several generalizations of the RS approach based on approximation spaces defined as pairs of the form (U, \mathcal{R}) , where \mathcal{R} is the equivalence relation (called indiscernibility relation) on the set U , have been reported in the literature [119, 120, 225, 248, 265–267, 274, 293, 313, 328, 330–332, 334]. Among extensions, not discussed in this chapter due to the space restriction, is the RS approach to multicriteria decision making (see, e.g., [60–65, 216, 288]).

Let us mention two of them.

A generalized approximation space^c can be defined by a tuple $\text{AS} = (U, \mathcal{I}, \nu)$ where \mathcal{I} is the *uncertainty function* defined on U with values in the powerset $\mathcal{P}(U)$ of U ($\mathcal{I}(x)$ is the *neighboorhood* of x) and ν is the *inclusion function* defined on the Cartesian product $\mathcal{P}(U) \times \mathcal{P}(U)$ with values in the interval $[0, 1]$ measuring the degree of inclusion of sets [261]. The lower and

^cMore general cases are considered, e.g., in articles [265, 267].

upper approximation operations can be defined in $\text{A\$}$ by

$$\text{LOW}_{\text{A\$}}(X) = \{x \in U : \nu(\mathcal{I}(x), X) = 1\}, \quad (11.12)$$

$$\text{UPP}_{\text{A\$}}(X) = \{x \in U : \nu(\mathcal{I}(x), X) > 0\}. \quad (11.13)$$

In the standard case, $\mathcal{I}(x)$ is equal to the equivalence class $B(x)$ of the indiscernibility relation $I\mathcal{N}\mathcal{D}_B$; in case of tolerance (similarity) relation $\mathcal{T} \subseteq U \times U$ we take $\mathcal{I}(x) = [x]_{\mathcal{T}} = \{y \in U : x \mathcal{T} y\}$, i.e., $\mathcal{I}(x)$ is equal to the tolerance class of \mathcal{T} defined by x . The standard rough inclusion relation ν_{SRI} is defined for $X, Y \subseteq U$ by

$$\nu_{SRI}(X, Y) = \begin{cases} \frac{|X \cap Y|}{|X|}, & \text{if } X \text{ is non-empty set,} \\ 1, & \text{otherwise.} \end{cases} \quad (11.14)$$

For applications it is important to have some constructive definitions of \mathcal{I} and ν .

One can consider another way to define $\mathcal{I}(x)$. Usually together with $\text{A\$}$ we consider some set \mathcal{F} of formulae describing sets of objects in the universe U of $\text{A\$}$ defined by semantics $\|\cdot\|_{\text{A\$}}$, i.e., $\|\alpha\|_{\text{A\$}} \subseteq U$ for any $\alpha \in \mathcal{F}^d$. Now, one can take the set

$$N_{\mathcal{F}}(x) = \{\alpha \in \mathcal{F} : x \in \|\alpha\|_{\text{A\$}}\}, \quad (11.15)$$

and $\mathcal{I}(x) = \{\|\alpha\|_{\text{A\$}} : \alpha \in N_{\mathcal{F}}(x)\}$. Hence, more general uncertainty functions having values in $\mathcal{P}(\mathcal{P}(U))$ can be defined and in the consequence different definitions of approximations are considered. For example, one can consider the following definitions of approximation operations in this approximation space $\text{A\$}$:

$$\text{LOW}_{\text{A\$o}}(X) = \{x \in U : \nu(Y, X) = 1 \text{ for some } Y \in \mathcal{I}(x)\}, \quad (11.16)$$

$$\text{UPP}_{\text{A\$o}}(X) = \{x \in U : \nu(Y, X) > 0 \text{ for any } Y \in \mathcal{I}(x)\}. \quad (11.17)$$

There are also different forms of rough inclusion functions. Let us consider two examples.

In the first example of a rough inclusion function, a threshold $t \in (0, 0.5)$ is used to relax the degree of inclusion of sets. The rough inclusion function ν_t is defined by

$$\nu_t(X, Y) = \begin{cases} 1 & \text{if } \nu_{SRI}(X, Y) \geq 1 - t, \\ \frac{\nu_{SRI}(X, Y) - t}{1 - 2t} & \text{if } t \leq \nu_{SRI}(X, Y) < 1 - t, \\ 0 & \text{if } \nu_{SRI}(X, Y) \leq t. \end{cases} \quad (11.18)$$

^dIf $\text{A\$} = (U, A)$ then we will also write $\|\alpha\|_U$ instead of $\|\alpha\|_{\text{A\$}}$.

This is an interesting “rough-fuzzy” example because we put the standard rough membership function as an argument into the formula often used for fuzzy membership functions.

One can obtain approximations considered in the variable precision rough set approach (VPRSM) [343] by substituting in (11.12)–(11.13) the rough inclusion function ν_t defined by (11.18) instead of ν , assuming that Y is a decision class and $\mathcal{I}(x) = B(x)$ for any object x , where B is a given set of attributes.

Another example of application of the standard inclusion was developed by using probabilistic decision functions. For more detail the reader is referred to [267, 283, 284, 309].

The rough inclusion relation can be also used for function approximation [265, 267, 274] and relation approximation [291]. In the case of function approximation the inclusion function ν^* for subsets $X, Y \subseteq U \times U$, where $U \subseteq \mathbb{R}$ and \mathbb{R} is the set of real numbers, is defined by

$$\nu^*(X, Y) = \begin{cases} \frac{|\pi_1(X \cap Y)|}{|\pi_1(X)|} & \text{if } \pi_1(X) \neq \emptyset, \\ 1 & \text{if } \pi_1(X) = \emptyset, \end{cases} \quad (11.19)$$

where π_1 is the projection operation on the first coordinate. Assume now, that X is a cube and Y is the graph $G(f)$ of the function $f : \mathbb{R} \rightarrow \mathbb{R}$. Then, e.g., X is in the lower approximation of f if the projection on the first coordinate of the intersection $X \cap G(f)$ is equal to the projection of X on the first coordinate. This means that the part of the graph $G(f)$ is “well” included in the box X , i.e., for all arguments that belong to the box projection on the first coordinate the value of f is included in the box X projection on the second coordinate. This approach was extended in several papers [267, 309].

The approach based on inclusion functions has been generalized to the *rough mereological approach* [194, 223, 224, 226]. The inclusion relation $x\mu_r y$ with the intended meaning x is a part of y to a degree at least r has been taken as the basic notion of the rough mereology being a generalization of the Leśniewski mereology [113, 114]. Research on rough mereology has shown importance of another notion, namely *closeness* of compound objects (e.g., concepts). This can be defined by $x cl_{r,r'} y$ if and only if $x \mu_r y$ and $y \mu_{r'} x$.

Rough mereology offers a methodology for synthesis and analysis of objects in a distributed environment of intelligent agents, in particular, for synthesis of objects satisfying a given specification to a satisfactory degree

or for control in such a complex environment. Moreover, rough mereology has been used for developing the foundations of the *information granule calculi*, aiming at formalization of the Computing with Words paradigm, recently formulated by Lotfi Zadeh [338]. More complex information granules are defined recursively using already defined information granules and their measures of inclusion and closeness. Information granules can have complex structures like classifiers or approximation spaces. Computations on information granules are performed to discover relevant information granules, *e.g.*, patterns or approximation spaces for compound concept approximations.

Usually there are considered families of approximation spaces labeled by some parameters. By tuning such parameters according to chosen criteria (*e.g.*, minimum description length) one can search for the optimal approximation space for concept description (see, *e.g.*, [12, 15, 163]).

Our knowledge about the approximated concepts is often partial and uncertain [66]. For example, concept approximation should be constructed from examples and counterexamples of objects for the concepts [81]. Hence, concept approximations constructed from a given sample of objects are extended, using inductive reasoning, on objects not yet observed.

The concept approximations should be constructed under dynamically changing environments [251, 273]. This leads to a more complex situation where the boundary regions are not crisp sets, which is consistent with the postulate of the higher order vagueness considered by philosophers [101]. Different aspects of vagueness in the rough set framework are discussed, *e.g.*, in the articles [136, 179, 180, 235, 251]. It is worthwhile to mention that a rough set approach to the approximation of compound concepts has been developed. For such concepts, it is hardly possible to expect that they can be approximated with the high quality by the traditional methods [26, 317]. The approach is based on hierarchical learning and ontology approximation [12, 172, 194, 259, 308]. Approximation of concepts in distributed environments is presented in [249]. A survey of algorithmic methods for concept approximation based on rough sets and boolean reasoning can be found in [12, 15, 163, 246].

11.5. Rough Sets and Induction

Granular formulas are constructed from atomic formulas corresponding to the considered attributes [209, 210, 265, 267]. In the consequence, the satisfiability of such formulas is defined if the satisfiability of atomic formulas

is given as the result of sensor measurement. Let us consider the two information systems $\mathbb{A} = (U, C, D)$ and $\mathbb{A}^* = (U^*, C)$ having the same set of attributes C , but $U \subseteq U^*$. Hence, one can consider for any constructed formula α over atomic formulas its semantics $\|\alpha\|_{\mathbb{A}} \subseteq U$ over U as well as the semantics $\|\alpha\|_{\mathbb{A}^*} \subseteq U^*$ over U^* .

The difference between these two cases is the following. In the case of U , one can compute $\|\alpha\|_{\mathbb{A}} \subseteq U$ but in the case $\|\alpha\|_{\mathbb{A}^*} \subseteq U^*$, for any object from $U^* - U$, there is no information about its membership relative to $\|\alpha\|_{\mathbb{A}^*} - \|\alpha\|_{\mathbb{A}}$. One can estimate the satisfiability of α for objects $u \in U^* - U$ only after the relevant sensory measurements on u are performed. In particular, one can use some methods for estimation of relationships among semantics of formulas over U^* using the relationships among semantics of these formulas over U . For example, one can apply statistical methods. This step is crucial in investigation of extensions of approximation spaces relevant for inducing classifiers from data.

The rough set approach is strongly related to inductive reasoning (*e.g.*, in rough set based methods for inducing classifiers or clusters [271]). The general idea for inducing classifiers is as follows. From a given decision table a set of granules in the form of decision rules is induced together with arguments *for* and *against* for each decision rule and decision class. For any new object with known signature one can select rules matching this object. Note that the left hand sides of decision rules are described by formulas making it possible to check for new objects if they satisfy them assuming that the signatures of these objects are known. In this way one can consider two semantics of formulas: on a sample of objects U and on its extension $U^* \supseteq U$. Definitely, one should consider a risk related to such generalization in the decision rule inducing. Next, a conflict resolution should be applied for resolving conflicts between matched rules by new object voting for different decisions. In the rough set approach, the process of inducing classifiers can be considered as the process of inducing approximations of concepts over extensions of approximation spaces (defined over samples of objects represented by decision systems). The whole procedure can be generalized for the case of approximation of more complex information granules. It is worthwhile mentioning that there were also developed approaches for inducing approximate reasoning schemes.

A typical approach in machine learning is based on inducing classifiers from samples of objects. These classifiers are used for prediction decisions on objects unseen so far, if only the signatures of these objects are available. This approach can be called global, *i.e.*, leading to decision extension from

a given sample of objects on the whole universe of objects. This global approach has some drawbacks (see Epilogue in [317]). Instead of this one can try to use transduction [317], semi-supervised learning, induced local models relative to new objects, or adaptive learning strategies. However, we are still far away from fully understanding discovery processes behind such generalization strategies [233].

11.5.1. Rough Sets and Classifiers

Rough sets are strongly related to inductive reasoning (*e.g.*, in rough set based methods for inducing classifiers or clusters).

In this section, we present an illustrative example of the rough set approach to induction of concept approximations. The approach can be generalized to the rough set approach to inductive extensions of approximation spaces.

Let us consider the problem of approximation of concepts over a universe U^∞ (concepts that are subsets of U^∞). We assume that the concepts are perceived only through some subsets of U^∞ , called samples. This is a typical situation in the machine learning, pattern recognition, or data mining approaches [33].

We assume that there is given an information system $\mathcal{A} = (U, A)$ and let us assume that for some $C \subseteq U^\infty$ there is given the set $\Pi_U(C) = C \cap U$. In this way we obtain a decision system $\mathbb{A}_d = (U, A, d)$, where $d(x) = 1$ if $x \in \Pi_U(C)$ and $d(x) = 0$, otherwise.

We would like to illustrate how from the decision function d may be induced a decision function μ_C defined over U^∞ with values in the interval $[0, 1]$ which can be treated as an approximation of the characteristic function of C .

Let us assume that $\text{RULES}(\mathbb{A}_d)$ is a set of decision rules induced by some rule generation method from \mathbb{A}_d . For any object $x \in U^\infty$, let $\text{MatchRules}(\mathbb{A}_d, x)$ be the set of rules from this set supported by x .

Now, the rough membership function $\mu_C : U^\infty \rightarrow [0, 1]$ approximating the characteristic function of C can be defined as follows

- (1) Let $R_k(x)$, for $x \in U^\infty$ be the set of all decision rules from $\text{MatchRules}(\mathbb{A}_d, x)$ with right hand side $d = k$, where $d = 1$ denotes that the rule r is voting for C and $d = 0$ – that the rule r is voting against C , respectively.
- (2) We define real values $w_k(x)$, where $w_1(x)$ is called the weight “for” and $w_0(x)$ the weight “against” membership of the object x in C , re-

spectively, by $w_k(x) = \sum_{r \in R_k(x)} strength(r)$, where $strength(r)$ is a normalized function depending on *length*, *support*, *confidence* of the decision rule r and on some global information about the decision system \mathcal{A}_d such as the size of the decision system or the class distribution.

- (3) Finally, one can define the value of $\mu_C(x)$ in the following way: $\mu_C(x)$ is undefined if $\max(w_1(x), w_0(x)) < \omega$; $\mu_C(x) = 0$ if $w_0(x) - w_1(x) \geq \theta$ and $w_0(x) > \omega$; $\mu_C(x) = 1$ if $w_1(x) - w_0(x) \geq \theta$ and $w_1(x) > \omega$ and $\mu_C(x) = \frac{\theta + (w_1(x) - w_0(x))}{2\theta}$, otherwise, where ω, θ are parameters set by user.

For computing of the value $\mu_C(x)$ for $x \in U^\infty$ the user should select a strategy resolving conflicting votes “for” and “against” membership of x in C . The degree of these conflicts are represented by values $w_1(x)$ and $w_0(x)$, respectively. Note that for some cases of x due to the small differences between these values the selected strategy may not produce the definite answer and these cases will create the boundary region.

We can now define the lower approximation, the upper approximation and the boundary region of the concept C relative to the induced rough membership function μ_C as follows

$$\begin{aligned} \text{LOW}_{\mu_C}(X) &= \{x \in U^\infty : \mu_C(x) = 1\}, \\ \text{UPP}_{\mu_C}(X) &= \{x \in U^\infty : \mu_C(x) > 0 \text{ or } \mu_C(x) \text{ is undefined}\}, \\ \text{BN}_{\mu_C}(X) &= \text{UPP}_{\mu_C}(X) \setminus \text{LOW}_{\mu_C}(X). \end{aligned} \quad (11.20)$$

The whole procedure can be generalized for the case of approximation of more complex information granules than concepts.

11.5.2. Inducing Relevant Approximation Spaces

A key task in Granular Computing (GrC) is the *information granulation* process that leads to the formation of information aggregates (with inherent patterns) from a set of available objects. A methodological and algorithmic issue is the formation of transparent (understandable) *information granules* inasmuch as they should provide a clear and understandable description of patterns present in sample objects [9, 211]. Such a fundamental property can be formalized by a set of constraints that must be satisfied during the information granulation process. For example, in case of inducing granules such as classifiers, the constraints specify requirements for the quality of classifiers. Then, inducing of classifiers can be understood as searching for relevant approximation spaces (which can be treated as a spacial type of granules) relative to some properly selected optimization measures.

Note that while there is a large literature on the covering based rough set approach (see, *e.g.*, [73, 342]) still much more work should be done on (scalable) algorithmic searching methods for relevant approximation spaces in huge families of approximation spaces defined by many parameters determining neighborhoods, inclusion measures and approximation operators. The selection of the optimization measures is not an easy task because they should guarantee that the (semi-) optimal approximation spaces selected relative to these criteria should allow us to construct classifiers of the high quality.

Let us consider some examples of optimization measures [164]. For example, the quality of an approximation space can be measured by:

$$\text{Quality}_1 : \mathcal{SAS}(U) \times \mathcal{P}(U) \rightarrow [0, 1], \quad (11.21)$$

where U is a non-empty set of objects and $\mathcal{SAS}(U)$ is a set of possible approximation spaces with the universe U .

Example 11.1. If $\text{UPP}_{\mathbb{A}\$}(X) \neq \emptyset$ for $\mathbb{A}\$ \in \mathcal{SAS}(U)$ and $X \subseteq U$ then

$$\text{Quality}_1(\mathbb{A}\$, X) = \nu_{SRI}(\text{UPP}_{\mathbb{A}\$}(X), \text{LOW}_{\mathbb{A}\$}(X)) = \frac{|\text{LOW}_{\mathbb{A}\$}(X)|}{|\text{UPP}_{\mathbb{A}\$}(X)|}. \quad (11.22)$$

The value $1 - \text{Quality}_1(\mathbb{A}\$, X)$ expresses the degree of completeness of our knowledge about X , given the approximation space $\mathbb{A}\$$.

Example 11.2. In applications, we usually use another quality measure analogous to the minimum description length principle [236, 237], where also the description length of approximation is included. Let us denote by $\text{description}(\mathbb{A}\$, X)$ the description length of approximation of X in $\mathbb{A}\$$. The description length may be measured, *e.g.*, by the sum of description lengths of algorithms testing membership for neighborhoods used in construction of the lower approximation, the upper approximation, and the boundary region of the set X . Then the quality $\text{Quality}_2(\mathbb{A}\$, X)$ can be defined by

$$\text{Quality}_2(\mathbb{A}\$, X) = g(\text{Quality}_1(\mathbb{A}\$, X), \text{description}(\mathbb{A}\$, X)), \quad (11.23)$$

where g is a relevant function used for fusion of values $\text{Quality}_1(\mathbb{A}\$, X)$ and $\text{description}(\mathbb{A}\$, X)$. This function g can reflect weights given by experts relative to both criteria.

One can consider different optimization problems relative to a given class $\mathcal{SAS}(U)$ of approximation spaces. For example, for a given $X \subseteq U$

and a threshold $t \in [0, 1]$, one can search for an approximation space AS satisfying the constraint $\text{Quality}_2(\text{AS}, X) \geq t$.

The reader is referred to [169] for more details on searching for relevant approximation spaces.

11.6. Discernibility and Boolean Reasoning: Rough Set Methods for Machine Learning, Pattern Recognition, and Data Mining

Tasks collected under the labels of data mining, knowledge discovery, decision support, pattern classification, and approximate reasoning require tools aimed at discovering *templates (patterns)* in data and classifying them into certain *decision classes*. Templates are in many cases most frequent sequences of events, most probable events, regular configurations of objects, the decision rules of high quality, standard reasoning schemes. Tools for discovery and classification of templates are based on *reasoning schemes* rooted in various paradigms [41]. Such patterns can be extracted from data by means of methods based, *e.g.*, on boolean reasoning and discernibility (see this section and [27]).

Discernibility relations belong to the most important relations considered in rough set theory. The ability to discern between perceived objects is important for constructing many entities like reducts, decision rules or decision algorithms. In the classical rough set approach, a discernibility relation $DIS(B) \subseteq U \times U$, where $B \subseteq A$ is a subset of attributes of an information system (U, A) , is defined by $xDIS(B)y$ if and only if $\text{non}(xIN\mathcal{D}_By)$, where $IN\mathcal{D}_B$ is the B -indiscernibility relation. However, this is, in general, not the case for the generalized approximation spaces.

The idea of boolean reasoning is based on construction for a given problem P of a corresponding boolean function f_P with the following property: The solutions for the problem P can be decoded from prime implicants of the boolean function f_P . Let us mention that to solve real-life problems it is necessary to deal with boolean functions having large number of variables.

A successful methodology based on discernibility of objects and boolean reasoning has been developed for computing of many entities important for applications, like reducts and their approximations, decision rules, association rules, discretization of real value attributes, symbolic value grouping, searching for new features defined by oblique hyperplanes or higher order surfaces, pattern extraction from data as well as conflict resolution or negotiation.

Most of the problems related to generation of the mentioned above entities are NP-complete or NP-hard. However, it was possible to develop efficient heuristics returning suboptimal solutions of the problems. The results of experiments on many data sets are very promising. They show very good quality of solutions generated by the heuristics in comparison with other methods reported in the literature (*e.g.*, with respect to the classification quality of unseen objects). Moreover, these heuristics are very efficient from the point of view of time necessary for computing of solutions. Many of these methods are based on discernibility matrices [260]. Note that it is possible to compute the necessary information about these matrices using^e information encoded in decision systems (*e.g.*, sorted in preprocessing [15, 173, 323]) directly, which significantly improves the efficiency of algorithms.

It is important to note that the methodology makes it possible to construct heuristics having a very important *approximation property* which can be formulated as follows: Expressions generated by heuristics, *i.e.*, implicants *close* to prime implicants define approximate solutions for the problem.

In supervised machine learning paradigm [81, 104, 129, 141], a learning algorithm is given a training data set, usually in the form of a decision system $\mathbb{A} = (U, A, d)^f$, prepared by an expert. Every such decision system classifies elements from U into decision classes. The purpose of the algorithm is to return a set of decision rules together with matching procedure and conflict resolution strategy, called a classifier, which makes it possible to classify unseen objects, *i.e.*, objects that are not described in the original decision table. In this section, we provide a number of rough set methods that can be used in construction of classifiers. For more information the reader is referred, *e.g.*, to [2, 18, 31, 32, 39, 40, 42, 54, 55, 58, 59, 68–72, 75, 76, 82, 86–88, 102], [7, 103, 110, 111, 115, 118, 122, 123, 142, 145, 158, 171, 188, 190–192, 194, 196, 206], [219, 223, 226, 227, 232, 239, 241, 242, 246, 256, 258, 292, 302–305, 307, 314, 320, 321, 324], and for papers on hierarchical learning and ontology approximation, *e.g.*, to [16, 19–21, 172, 174, 175, 249, 259, 263, 264].

Many of the developed techniques are based on computing prime implicants of boolean functions for computing different kinds of reducts. Unfortunately, they are computationally hard. However, many heuristics have been developed which turned out to be very promising. The results of ex-

^eThat is, without the necessity of generation and storing of the discernibility matrices

^fFor simplicity, we consider decision systems with one decision.

periments on many data sets, reported in the literature, show a very good quality of classification of unseen objects using these heuristics. A variety of methods for computing reducts and their applications can be found in [15, 109, 121, 163, 196, 226, 227, 246, 247, 258, 260, 279, 284, 324, 325]. The fact that the problem of finding a minimal reduct of a given information system is NP-hard was proved in [260].

As we mentioned, there exists a number of good heuristics that compute sufficiently many reducts in an acceptable time. Moreover, a successful methodology, based on different reducts, has been developed for solution of many problems like attribute selection, decision rule generation, association rule generation, discretization of real-valued attributes, and symbolic value grouping. For further readings the reader is referred to [18, 163, 305] (attribute selection); [159, 160, 163, 165, 167, 255] (discretization); [161–163] (discretization of data stored in relational databases); and [163, 166] (reduct approximation and association rules) and survey articles [163, 169, 169, 272].

Many of these methods are based on discernibility matrices [260]. It is worthwhile to mention that it is possible to compute the necessary information about these matrices directly from information or decision systems (*e.g.*, sorted in preprocessing [15, 173]) what significantly improves the efficiency of algorithms (see also Section 11.7).

Several methods based on boolean reasoning have been implemented in the Rough Set Exploration System (RSES): <http://logic.mimuw.edu.pl/~rses/> (see also [14, 15, 17, 22, 105]). For links to other rough set software systems the reader is referred to the Rough Set Database System (RSDS): <http://rsds.univ.rzeszow.pl/home>.

For more readings on the rough set based methods for features selection the reader is referred to surveys [97, 240] and articles [5, 8, 83, 85, 137, 140, 163, 169, 230, 231, 295, 305, 318, 319, 335].

There have been also developed methods for approximation of compound concepts based on rough sets, hierarchical learning, and ontology approximation (see, *e.g.*, [12, 16, 19–21, 172, 174, 175, 249, 259, 263, 264]).

11.7. Scalable Rough Set Methods

One of the central issues in research based on rough sets concerns development of the scalable methods. In this section we discuss two examples.

Let us illustrate the idea of reduct calculation using discernibility matrix [260] (Table 11.2).

Example 11.3. Let us consider the “weather” problem, which is represented by decision table (see Table 11.1). Objects are described by four conditional attributes and are divided into 2 classes. Let us consider the first 12 observations. In this example, $U = \{1, 2, \dots, 12\}$, $A = \{a_1, a_2, a_3, a_4\}$, $CLASS_{no} = \{1, 2, 6, 8\}$, $CLASS_{yes} = \{3, 4, 5, 7, 9, 10, 11, 12\}$.

Table 11.1. The exemplary “weather” decision table.

date	outlook	temperature	humidity	windy	play
ID	a_1	a_2	a_3	a_4	dec
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes

Table 11.2. The compact form of discernibility matrix, where in the entry i, j is the set of all attributes (more precisely, propositional variables corresponding to them, for simplicity denoted in the same way as attributes) discerning objects x_i and x_j .

\mathcal{M}	1	2	6	8
3	a_1	a_1, a_4	$a_1, a_2,$ a_3, a_4	a_1, a_2
4	a_1, a_2	a_1, a_2, a_4	a_2, a_3, a_4	a_1
5	a_1, a_2, a_3	$a_1, a_2,$ a_3, a_4	a_4	a_1, a_2, a_3
7	$a_1, a_2,$ a_3, a_4	a_1, a_2, a_3	a_1	$a_1, a_2,$ a_3, a_4
9	a_2, a_3	a_2, a_3, a_4	a_1, a_4	a_2, a_3
10	a_1, a_2, a_3	$a_1, a_2,$ a_3, a_4	a_2, a_4	a_1, a_3
11	a_2, a_3, a_4	a_2, a_3	a_1, a_2	a_3, a_4
12	a_1, a_2, a_4	a_1, a_2	a_1, a_2, a_3	a_1, a_4

The discernibility matrix can be treated as a board containing $n \times n$ boxes. Noteworthy is the fact that discernibility matrix is symmetrical with respect to the main diagonal, because $\mathcal{M}_{i,j} = \mathcal{M}_{j,i}$, and that sorting all objects according to their decision classes causes a shift off all empty boxes nearby to the main diagonal. In case of decision table with two decision classes, the discernibility matrix can be rewritten in a more compact form as shown in Table 11.2. The discernibility function is constructed from

discernibility matrix by taking a conjunction of all discernibility clauses. After reducing of all repeated clauses we have:

$$\begin{aligned} f(a_1, a_2, a_3, a_4) \equiv & (a_1) \wedge (a_1 \vee a_4) \wedge (a_1 \vee a_2) \wedge (a_1 \vee a_2 \vee a_3 \vee a_4) \wedge \\ & (a_1 \vee a_2 \vee a_4) \wedge (a_2 \vee a_3 \vee a_4) \wedge (a_1 \vee a_2 \vee a_3) \wedge \\ & (a_4) \wedge (a_2 \vee a_3) \wedge (a_2 \vee a_4) \wedge (a_1 \vee a_3) \wedge (a_3 \vee a_4) \wedge \\ & (a_1 \vee a_2 \vee a_4). \end{aligned}$$

One can find relative reducts of the decision table by searching for its prime implicants. The straightforward method calculates all prime implicants by translation to DNF (using absorbtion rule $p(p + q) \equiv p$ another rules for boolean algebra). One can do it as follow:

$$f \equiv (a_1) \wedge (a_4) \wedge (a_2 \vee a_3) = a_1 \wedge a_4 \wedge a_2 \vee a_1 \wedge a_4 \wedge a_3.$$

Thus we have 2 reducts: $R_1 = \{a_1, a_2, a_4\}$ and $R_2 = \{a_1, a_3, a_4\}$.

Every heuristic for the prime implicant problem can be applied to the discernibility function to solve the minimal reduct problem. One of such heuristics was proposed in [260] and was based on the idea of greedy algorithm, where each attribute is evaluated by its discernibility measure, *i.e.*, the number of pairs of objects which are discerned by the attribute, or, equivalently, the number of its occurrences in the discernibility matrix.

- First we have to calculate the number of occurrences of each attributes in the discernibility matrix:

$$\begin{array}{ll} eval(a_1) = disc_{dec}(a_1) = 23 & eval(a_2) = disc_{dec}(a_2) = 23 \\ eval(a_3) = disc_{dec}(a_3) = 18 & eval(a_4) = disc_{dec}(a_4) = 16 \end{array}$$

Thus a_1 and a_2 are the two most preferred attributes.

- Assume that we select a_1 . Now we are taking under consideration only those cells of the discernibility matrix which are not containing a_1 . There are 9 such cells only, and the number of occurrences are as the following:

$$\begin{array}{l} eval(a_2) = disc_{dec}(a_1, a_2) - disc_{dec}(a_1) = 7 \\ eval(a_3) = disc_{dec}(a_1, a_3) - disc_{dec}(a_1) = 7 \\ eval(a_4) = disc_{dec}(a_1, a_4) - disc_{dec}(a_1) = 6 \end{array}$$

- If this time we select a_2 , then there are only 2 remaining cells, and both are containing a_4 ;

- Therefore, the greedy algorithm returns the set $\{a_1, a_2, a_4\}$ as a reduct of sufficiently small size.

There is another reason for choosing a_1 and a_4 , because they are *core attributes*.^g It has been shown that an attribute is a core attribute if and only if occurs in the discernibility matrix as a singleton [260]. Therefore, core attributes can be recognized by searching for all single cells of the discernibility matrix. The pseudo-code of the heuristic searching for a short reduct is presented in Algorithm 11.1.

Algorithm 11.1 (Searching for short reduct).

begin

$B := \emptyset;$

// Step 1. Initializing B by core attributes

for $a \in A$ **do**

if $isCore(a)$ **then**

$B := B \cup \{a\};$

end

end

// Step 2. Including attributes to B

repeat

$a_{max} := \arg \max_{a \in A - B} disc_{dec}(B \cup \{a\});$

$eval(a_{max}) := disc_{dec}(B \cup \{a_{max}\}) - disc_{dec}(B);$

if $eval(a_{max}) > 0$ **then**

$B := B \cup \{a\};$

end

until $(eval(a_{max}) == 0) \text{ OR } (B == A);$

// Step 3. Elimination

for $a \in B$ **do**

if $disc_{dec}(B) = disc_{dec}(B - \{a\})$ **then**

$B := B - \{a\};$

end

end

end

The reader may have a feeling that the greedy algorithm for reduct problem has quite a high complexity, because two main operations:

^gAn attribute is called core attribute if and only if it occurs in every reduct.

- $disc(B)$ – number of pairs of objects discerned by attributes from B ;
- $isCore(a)$ – check whether a is a core attribute;

are defined by the discernibility matrix which is a complex data structure containing $O(n^2)$ cells, and each cell can contain up to $O(m)$ attributes, where n is the number of objects and m is the number of attributes of the given decision table. This suggests that the two main operations need at least $O(mn^2)$ computational time.

Fortunately, both operations can be performed more efficiently. It has been shown [173] that both operations can be calculated in time $O(mn \log n)$ without the necessity to store the discernibility matrix.

Now we will show that this algorithm can be efficiently implemented in DBMS using only simple SQL queries.

Let $\mathbb{A} = (U, A, dec)$ be a decision table. By “*counting table*” of a set of objects $X \subset U$ we denote the vector:

$$CountTable(X) = \langle n_1, \dots, n_d \rangle,$$

where $n_k = card(X \cap CLASS_k)$ is the number of objects from X belonging to the k^{th} decision class. We define a conflict measure of X by

$$conflict(X) = \sum_{i < j} n_i n_j = \frac{1}{2} \left[\left(\sum_{k=1}^d n_k \right)^2 - \sum_{k=1}^d n_k^2 \right].$$

In other words, $conflict(X)$ is the number of pairs of different class objects in X .

By *counting table* of a set of attributes B we mean the two-dimensional array $Count(B) = [n_{v,k}]_{v \in INF(B), k \in V_{dec}}$, where

$$n_{v,k} = card(\{x \in U : inf_B(x) = v \text{ and } dec(x) = k\}).$$

Thus $Count(B)$ is a collection of counting tables of equivalence classes of the indiscernibility relation $I\mathcal{ND}_B$. It is clear that the complexity time for the construction of counting table is $O(nd \log n)$, where n is the number of objects and d is the number of decision classes. It is clear that counting table can be easily constructed in data base management systems using simple SQL queries.

The discernibility measure of a set of attributes B can be easily calculated from the counting table as follows:

$$disc_{dec}(B) = \frac{1}{2} \sum_{v \neq v', k \neq k'} n_{v,k} \cdot n_{v',k'}.$$

The disadvantage of this equation relates to the fact that it requires $O(S^2)$ operations, where S is the size of the counting table $Count(B)$.

The discernibility measure can be understood as a number of unresolved (by the set of attributes B) conflicts. One can show that:

$$disc_{dec}(B) = conflict(U) - \sum_{[x] \in U / I\mathcal{N}\mathcal{D}_B} conflict([x]_{I\mathcal{N}\mathcal{D}_B}). \quad (11.24)$$

Thus, the discernibility measure can be determined in $O(S)$ time:

$$disc_{dec}(B) = \frac{1}{2} \left(n^2 - \sum_{k=1}^d n_k^2 \right) - \frac{1}{2} \sum_{v \in INF(B)} \left[\left(\sum_{k=1}^d n_{v,k} \right)^2 - \sum_{k=1}^d n_{v,k}^2 \right], \quad (11.25)$$

where $n_k = |CLASS_k| = \sum_v n_{v,k}$ is the size of k^{th} decision class.

Moreover, one can show that attribute a is a core attribute of decision table $= (U, A \cup \{dec\})$ if and only if

$$disc_{dec}(A - \{a\}) < disc_{dec}(A).$$

Thus both operations $disc_{dec}(B)$ and $isCore(a)$ can be performed in linear time with respect to the counting table.

Example 11.4. The counting table for a_1 is as follows:

$Count(a_1)$	$dec = no$	$dec = yes$
$a_1 = sunny$	3	2
$a_1 = overcast$	0	3
$a_1 = rainy$	1	3

We illustrate Eq. (11.25) by inserting some additional columns to the counting table:

$Count(a_1)$	$dec = no$	$dec = yes$	\sum	$conflict(.)$
$a_1 = sunny$	3	2	5	$\frac{1}{2}(5^2 - 2^2 - 3^2) = 6$
$a_1 = overcast$	0	3	3	$\frac{1}{2}(3^2 - 0^2 - 3^2) = 0$
$a_1 = rainy$	1	3	4	$\frac{1}{2}(4^2 - 1^2 - 3^2) = 3$
U	4	8	12	$\frac{1}{2}(12^2 - 8^2 - 4^2) = 32$

Thus $disc_{dec}(a_1) = 32 - 6 - 0 - 3 = 23$.

Another scalable method for generation of reducts is presented in [108]. The reader is also referred to the book [134] for the rough set based scalable methods with applications in bioinformatics, to [18] for application of *dynamic reducts*, to [107] for application of *random reducts* to the feature selection problem, to [327] parallel attribute reduction with application of MapReduce or to [106] for scalable rough set based methods using FPGA.

Mining large data sets is one of the biggest challenges in Knowledge Discovery and Data Mining. In many practical applications, there is a need of data mining algorithms running on terminals of possibly distributed database systems where the only access to data is enabled by SQL queries or NoSQL operations.

Let us consider two illustrative examples of problems for large data sets: (i) searching for short reducts, (ii) searching for best partitions defined by cuts on continuous attributes. In both cases the traditional implementations of rough sets and boolean reasoning based methods are characterized by the high computational cost. The critical factor for time complexity of algorithms solving the discussed problems is the number of data access operations. Fortunately some efficient modifications of the original algorithms were proposed by relying on concurrent retrieval of higher level statistics which are sufficient for the heuristic search of reducts and partitions (see, e.g., [161–163, 271]). Numerous experiments on different data sets have shown that the proposed solutions allows one to find a cut which is very close to the optimal one. For more details the reader is referred to the literature (see [161, 162]).

One of the great commercial application of the rough set approach is related to other scalable big data processing techniques (see e.g., Infobright <http://www.infobright.com/>, [281, 282]).

11.8. Hybridization of Rough Sets, Fuzzy Sets, Neural Networks, and Other Approaches

Both fuzzy and rough set theory represent two different approaches to vagueness. Fuzzy set theory addresses *gradualness* of knowledge, expressed by the fuzzy membership, whereas rough set theory addresses *granularity* of knowledge, expressed by the indiscernibility relation. Both theories are not competing but are rather complementary. In particular, the rough set approach provides tools for approximate construction of fuzzy membership functions.

Rough sets and fuzzy sets can work synergistically, often with other soft computing approaches. The developed systems exploit the tolerance for imprecision, uncertainty, approximate reasoning and partial truth under soft computing framework and is capable of achieving tractability, robustness, and close resemblance with human like (natural) decision making for pattern recognition in ambiguous situations [257, 337]. The developed methods have found applications in different domains such as bioinfor-

matics and medical image processing. The objective of the rough-fuzzy integration is to provide a stronger paradigm of uncertainty handing in decision-making. Over the year many methods and applications, in particular in pattern recognition were developed on the basis of rough sets or fuzzy sets and on their combination. The methods based on combination of the approaches exploit different abilities of mixed languages used for generation and expressing patterns by both approaches based on rough set and fuzzy sets, respectively. This is making it possible to discover patterns of the higher quality in comparison with situations when they are used in isolation due to better possibility of approximation of the boundary regions of vague concepts. One should note that unfortunately in this case the searching space for relevant patters is becoming larger in comparison with cases when single approach is used and developing efficient heuristics searching for relevant patterns is more challenging. The developed methods concern discovery of patterns such as decision rules, clusters and processes or feature selection. The reader can find more details in the literature (see, *e.g.*, [124, 163, 256]) for the rough set based methods and [23, 50, 51, 98, 99, 130–133, 182, 183, 186, 191, 193, 196, 213, 239, 257], [34, 79, 238]) dedicated to methods based on combination of rough sets and fuzzy sets. There is also a substantial progress in understanding theoretical foundations of relationships between rough sets and fuzzy sets (see, *e.g.*, [90]). The further research will certainly lead to new more efficient methods based on combination of rough sets and fuzzy sets.

The characteristics of rough-fuzzy granulation have been further exploited in designing various neural network models for their efficient and speedy learning, and enhanced performance (see, *e.g.*, [6, 7, 48–51, 122, 142, 143, 188, 190, 192, 194]). This has a strong promise to Big data analysis, recently drawn the attention of researchers world-wide.

There are hybrid methods combining rough sets with methods using among others statistics, kernel functions, case-based reasoning, wavelets, EM method, independent component analysis, principal component analysis (see, *e.g.*, [1, 3, 84, 118, 138, 144, 185, 189, 307]).

11.9. Interactive (Rough) Granular Computing (I(R)GrC)

In this section, we start from a discussion on information granulation and we outline an approach to interactive computations based on complex granules (c-granules, for short). We emphasize the need for special reasoning called *adaptive judgment* on interactive computations over c-granules. This

reasoning is also important for discovery of complex patterns in interactive computations on complex granules. We claim that the approach can be used for developing methods for approximation of complex vague concepts and approximate reasoning about them. In the following section, we shortly describe the approximation ontology approach. Next, we outline three other important challenging topics related to I(R)GrC.

11.9.1. *Information Granulation*

Information granulation can be viewed as a human way of achieving data compression and it plays a key role in the implementation of the strategy of divide-and-conquer in human problem-solving [211, 338]. Objects obtained as the result of granulation are information granules. Such granules are obtained as the result of information granulation [338]:

Information granulation can be viewed as a human way of achieving data compression and it plays a key role in implementation of the strategy of divide-and-conquer in human problem-solving.

Granulation is inherent in human thinking and reasoning processes. Granular computing (GrC) provides an information processing framework where computation and operations are performed on information granules, and it is based on the realization that precision is sometimes expensive and not much meaningful in modeling and controlling complex systems.

Examples of elementary information granules are indiscernibility or tolerance (similarity) classes (see, e.g., [210]). In reasoning about data and knowledge under uncertainty and imprecision many other more compound information granules are used (see, e.g., [228, 229, 248, 262, 263]). Examples of such granules are clusters, decision rules, sets of decision rules or classifiers. More compound information granules are defined by means of less compound ones. Note that inclusion or closeness measures between information granules should be considered rather than their strict equality. Such measures are also defined recursively for information granules.

When a problem involves incomplete, uncertain, and vague information, it may be difficult to differentiate distinct elements and one may find it convenient to consider granules for its handling. The structure of granulation can be often defined using methods based on rough sets, fuzzy sets or their combination. In this consortium, rough sets and fuzzy sets work synergistically, often with other soft computing approaches, and use the principles

of granular computing [257, 337] (see Sect. 11.8).

Constructions of information granules should be robust with respect to their input information granule deviations. In this way, also a granulation of information granule constructions is considered. As the result we obtain the so called (approximate reasoning schemes (AR schemes or AR networks) [194, 223, 224, 226]. AR schemes can be interpreted as complex patterns [104]. Searching methods for such patterns relevant for a given target concept have been developed [194]. Methods for deriving relevant AR schemes are of high computational complexity. The complexity can be substantially reduced by using domain knowledge. In such a case AR schemes are derived along reasoning schemes in natural language that are retrieved from domain knowledge. Developing methods for deriving such AR schemes is one of the main goals of our current projects.

11.9.2. *I(R)GrC Foundations: Interactive Computations on Complex Granules*

Computations on granules should be interactive. This requirement is fundamental for modeling of complex systems [57]. For example, this is expressed by the following sentence [178]:

[...] interaction is a critical issue in the understanding of complex systems of any sorts: as such, it has emerged in several well-established scientific areas other than computer science, like biology, physics, social and organizational sciences.

Granular Computing (GrC) was recently extended to I(R)GrC by introducing *complex granules* (*c-granules*) and interactive computations based on c-granules. We propose to model (complex) intelligent systems by societies of agents [91, 94, 95].

I(R)GrC is an approach for modeling interactive computations [91, 94, 95, 275, 276, 276]. I(R)GrC are progressing by interactions between granules (structural objects of quite often high order type) discovered from data and domain knowledge. In particular, interactive information systems (IIS) are dynamic granules used for representing the results of the agent interaction with the environments. IIS can be also applied in modeling more advanced forms of interactions such as hierarchical interactions in layered granular networks or generally in hierarchical modeling.

Any c-granule consists of three components, namely soft_suit, link_suit and hard_suit. These components are making it possible to deal with such

abstract objects from soft_suit as infogranules as well as with physical objects from hard_suit. The link_suit of a given c-granule is used as a kind of c-granule interface for handling interaction between soft_suit and hard_suit [91].

Calculi of c-granules are defined beginning from some elementary c-granules (corresponding to, *e.g.*, reading or storing measured values, simple sensory measurements, indiscernibility or similarity classes). Next the agent control as well as interactions with the environment are making it possible to generate new c-granules from the already defined ones. The hard_suits, link_suits, and soft_suits of more compound c-granules are defined using the relevant networks over already defined c-granules. The networks are satisfying some constraints which can be interpreted as definitions of types of networks. The link_suits of such more compound granules are responsible for transmission of interactions between the hard_suits and soft_suits represented by the corresponding networks. The results and/or properties of transmitted interactions are recorded in the soft_suits. We assume that information about states of (some) physical beings from soft_suit can be decoded using expressions from the agent language (*e.g.*, consisting of propositional formula over descriptors [204, 210] or expressions from a (simplified) fragment of natural language [338]). Moreover, a c-granule is making it possible to record in its soft_suit the perceived by it interactions in its hard_suit which are transmitted by the link_suit to the soft_suit. This is typical for sensory measurement. On the other hand, a c-granule may cause some interactions in its hard_suit by transmitting through its link_suit some interactions from the soft_suit. However, the c-granule may perceive the results (or properties) of such caused in the hard_suit interactions only by using the soft_suit. This is done on the basis of the transmitted results (or properties) of these caused interactions in the hard_suit by transmitting them back through the link_suit to the soft_suit. These results (or properties) may be different from the predicted ones which can be a priori stored in soft_suit. This is typical for performing of actions initiated by c-granules.

Any agent operates in a local world of c-granules. The agent control is aiming at controlling computations performed on c-granules from this local world for achieving the target goals. Actions (sensors or plans) from link_suits of c-granules are used by the agent control in exploration and/or exploitation of the environment on the way to achieve their targets. C-granules are also used for representation of perception by agents concerning of interactions in the physical world. Due to the bounds of the agent perception abilities usually only a partial information about the interactions

from physical world may be available for agents. Hence, in particular the results of performed actions by agents can not be predicted with certainty. For more details on I(R)GrC based on c-granules the reader is referred to [91].

We assume that for any agent there is a distinguished family of her/his c-granules creating the *internal language* of the agent [91]. We assume that elements of the internal language can be encoded by information granules (or, infogranules for short).

One should note that the process of (discovering) distinguishing the relevant family of c-granules creating the internal language is a very complex process. The relevant infogranules are discovered in hierarchical aggregation of infogranules considered in relevant contexts. In general, such infogranules are called semiotic c-granules [91]. Infogranules are used for constructing the target infogranules. On the basis of satisfiability (to a degree) of such target infogranules (interpreted as approximations of complex vague concepts) relevant actions are undertaken by the agent aiming to satisfy her/his needs. An example of the agent internal language can be defined by c-granules representing propositional formulas over descriptors [204, 210].

11.9.3. Toward Approximation of Complex Vague Concepts and Reasoning about Them

There are many real-life problems that are still hard to solve using the existing methodologies and technologies. Among such problems are, *e.g.*, classification and understanding of medical images, control of autonomous systems like unmanned aerial vehicles or robots, problems related to monitoring or rescue tasks in multiagent systems. All of these problems are closely related to intelligent systems that are more and more widely applied in different real-life projects.

One of the main problems investigated in machine learning, pattern recognition [81] and data mining [104] is concept approximation. It is necessary to induce approximations of concepts (models of concepts) from available experimental data. The data models developed so far in such areas like statistical learning, machine learning, pattern recognition are not satisfactory for approximation of compound concepts resulting in the perception process. Researchers from the different areas have recognized the necessity to work on new methods for concept approximation (see, *e.g.*, [26, 317]). The main reason is that these compound concepts are, in a sense, *too far*

from measurements which makes the searching for relevant (for their approximation) features infeasible in a huge space. There are several research directions aiming at overcoming this difficulty. One of them is based on the interdisciplinary research where the results concerning perception in psychology or neuroscience are used to help to deal with compound concepts (see, *e.g.*, [81]). There is a great effort, *e.g.*, in neuroscience towards understanding the hierarchical structures of neural networks in living organisms [43, 220]. Also mathematicians are recognizing problems of learning as the main problem of the current century [220]. The problems discussed so far are also closely related to complex system modeling. In such systems again the problem of concept approximation and reasoning about perceptions using concept approximations is one of the challenges nowadays. One should take into account that modeling complex phenomena entails the use of local models (captured by local agents, if one would like to use the multi-agent terminology [89, 341]) that next should be fused [311]. This process involves the negotiations between agents [89] to resolve contradictions and conflicts in local modeling. This kind of modeling will become more and more important in solving complex real-life problems which we are unable to model using traditional analytical approaches. The latter approaches lead to exact models. However, the necessary assumptions used to develop them are causing the resulting solutions to be too far from reality to be accepted. New methods or even a new science should be developed for such modeling [56].

One of the possible solutions in searching for methods for compound concept approximations is the layered learning idea [294]. Inducing concept approximation should be developed hierarchically starting from concepts close to sensor measurements to compound target concepts related to perception. This general idea can be realized using additional domain knowledge represented in natural language. For example, one can use principles of behavior on the roads, expressed in natural language, trying to estimate, from recordings (made, *e.g.*, by camera and other sensors) of situations on the road, if the current situation on the road is safe or not. To solve such a problem one should develop methods for concept approximations together with methods aiming at approximation of reasoning schemes (over such concepts) expressed in natural language. Foundations of such an approach are based on rough set theory [204] and its extension rough mereology [194, 223, 224, 226], both discovered in Poland.

Qualitative reasoning requires to develop methods supporting approximate reasoning under uncertainty about non-crisp concepts, often vague

concepts. One of the very general scheme of tasks for such qualitative reasoning can be described as follows. From some basic objects (called in different areas as patterns, granules or molecules) it is required to construct (induce) complex objects satisfying a given specification (often, expressed in natural language specification) to a satisfactory degree. For example, in learning concepts from examples we deal with tasks where a partial information about the specification is given by examples and counter examples concerning of classified objects. As examples of such complex objects one can consider classifiers considered in Machine Learning or Data Mining, new medicine against some viruses or behavioral patterns of cell interaction induced from interaction of biochemical processes realized in cells. Over the years we have learned how to solve some of such tasks while many of them are still challenges. One of the reasons is that the discovery process of complex objects relevant for the given specification requires multilevel reasoning with necessity of discovering on each level the relevant structural objects and their properties. The searching space for such structural objects and properties is very huge and this, in particular, causes that fully automatic methods are not feasible using the exiting computing technologies. However, this process can be supported by domain knowledge used which can be used for generating hints in the searching process (see, e.g., [12]). This view is consistent with [25] (see, page 3 of Foreword):

[...] Tomorrow, I believe, every biologist will use computer to define their research strategy and specific aims, manage their experiments, collect their results, interpret their data, incorporate the findings of others, disseminate their observations, and extend their experimental observations - through exploratory discovery and modeling - in directions completely unanticipated.

In one of the previous sections we presented a view that the computation model on which the searching for relevant patterns – for approximation of the discussed above complex vague concepts – should be based is a model of interactive computations based on complex granules (c-granules, for short).

11.9.4. *Ontological Framework for Approximation*

We have discovered that hierarchical modeling is required for approximation of complex vague concepts [176, 220]. In a number of papers [12, 13, 172, 187, 264], the problem of ontology approximation has been discussed together with possible applications to approximation of compound

concepts or to knowledge transfer [12, 13, 16, 172, 175, 250, 252, 264]. For software RoughICE supporting ontology approximation the reader is referred to the system homepage^h.

In the ontology [289] (vague) concepts and local dependencies between them are specified. Global dependencies can be derived from local dependencies. Such derivations can be used as hints in searching for relevant compound patterns (information granules) in approximation of more compound concepts from the ontology. The ontology approximation problem is one of the fundamental problems related to approximate reasoning in distributed environments. One should construct (in a given language that is different from the ontology specification language) not only approximations of concepts from ontology but also vague dependencies specified in the ontology. It is worthwhile to mention that an ontology approximation should be induced on the basis of incomplete information about concepts and dependencies specified in the ontology. Information granule calculi based on rough sets have been proposed as tools making it possible to solve this problem. Vague dependencies have vague concepts in premisses and conclusions. The approach to approximation of vague dependencies based only on degrees of closeness of concepts from dependencies and their approximations (classifiers) is not satisfactory for approximate reasoning. Hence, more advanced approach should be developed. Approximation of any vague dependency is a method which allows for any object to compute the arguments “for” and “against” its membership to the dependency conclusion on the basis of the analogous arguments relative to the dependency premisses. Any argument is a compound information granule (compound pattern). Arguments are fused by local schemes (production rules) discovered from data. Further fusions are possible through composition of local schemes, called approximate reasoning schemes (AR schemes) [21, 194, 228]. To estimate the degree to which (at least) an object belongs to concepts from ontology the arguments “for” and “against” those concepts are collected and next a conflict resolution strategy is applied to them to predict the degree.

^h<http://www.mimuw.edu.pl/~bazan/roughice/>

11.9.5. Context Inducing

One of the old and still challenging problem in machine learning, pattern recognition and data mining is feature discovery (feature constructive induction, feature extraction) [81]. Dealing with this problem, one should first induce relevant structures of objects over which the features are defined. Such structures can be treated as contexts of the considered objects. We would like to emphasize that information systems may be used for context modeling. The approach is based on fusion (aggregation) of information systems (or decision tables) with constraints [275]. The constraints can be defined by means of relations over sets of attribute values or their Cartesian products. Objects on the next level of modeling are relational structures over signatures of objects defined by constraints. In this way, one can obtain as objects on succeeding hierarchical levels such more complex objects as indiscernibility (similarity) classes of objects, time windows, their clusters, sequences of time windows or their sets. Indiscernibility classes over objects representing sequences of time windows are sets of such sequences and they may represent information about processes.

Discovery of relevant relations between object signatures is an important step in searching for relevant approximation spaces. In this way, a class of relational structures representing perception of objects and their parts is constructed. In the next step, we select a language \mathcal{L} consisting of formulas expressing properties over the defined relational structures and we search for relevant formulas in \mathcal{L} . The semantics of formulas (*e.g.*, with one free variable) from \mathcal{L} are subsets of object from the considered hierarchical level of modeling.

Quite often this searching process for relevant structures of objects and their features is sophisticated. For example, one can discover several relational structures (*e.g.*, corresponding to different attributes) and formulas over such structures defining different families of neighborhoods from the original approximation space. As a next step, such families of neighborhoods can be merged into neighborhoods in a new, higher degree approximation space.

The proposed approach is making it possible to construct information systems (or decision tables) on a given level of hierarchical modeling from information systems from lower level(s) by using some constraints in joining objects from underlying information systems [275]. In this way, structural objects can be modeled and their properties can be expressed in constructed information systems by selecting relevant attributes. These attributes are

defined with use of a language that makes use of attributes of systems from the lower hierarchical level as well as relations used to define constraints. In some sense, the objects on the next level of hierarchical modeling are defined using the syntax from the lower level of the hierarchy. Domain knowledge is used to aid the discovery of relevant attributes (features) on each level of hierarchy. This domain knowledge can be provided, *e.g.*, by concept ontology together with samples of objects illustrating concepts from this ontology. Such knowledge is making it feasible to search for relevant attributes (features) on different levels of hierarchical modeling.

11.9.6. Process Mining

In the data mining community, there is a rapidly growing interest in developing methods for process mining, *e.g.*, for discovery of structures of temporal processes from observations (recorded data). Works on process mining [24, 36, 315, 316, 326] have recently been undertaken by many renowned centers worldwide¹. This research is also related, *e.g.*, to functional data analysis [234], cognitive networks [201], and dynamical system modeling in biology [44].

There are numerous papers on discovery of concurrent processes from data based on rough sets [37, 152, 153, 155, 156, 164, 197–200, 217, 218, 268–270, 296–301, 306]. In papers [164, 168], is outlined an approach to discovery of processes from data and domain knowledge which is based on IRGrC philosophy. This research was initiated by the idea of Professor Zdzisław Pawlak [205]. Data tables are treated as partial specifications of concurrent processes. Rows in a given data table are examples of global states and columns represent local processes. One of the solutions presented in the above papers was based on decomposition of data tables into modules defined by reducts of data tables. The modules are linked by constraints defined by rules extracted from data. In another approach, first from a given data table decision rules are extracted (*e.g.*, a set of minimal decision rules) and such a set of decision rules is used as knowledge encoded in the data table or theory defined by data table. Next, the set of all global states is defined as equal to the maximal set of objects (global states) consistent with the theory. There were proposed methods for automatic generation from a given data table a (colored) Petri net with the reachability set equal to the maximal consistent set of states consistent with the theory

¹<http://www.isle.org/~langley/>, <http://soc.web.cse.unsw.edu.au/bibliography/discovery/index.html>, <http://www.processmining.org/>

generated from the data table. The reader is referred to the Web page <http://rsds.univ.rzeszow.pl/home> for information on the developed software (ROSECON) for inducing Petri nets from data tables. An important role in discovering Petri nets play the inhibitory rules [37]. The reader interested in complexity results related to such rules is referred to [37].

Here, we would like to formulate some challenges related to discovery of concurrent processes from data tables occurring in hierarchical modeling by using I(R)GrC. On higher levels of hierarchy, the structure of objects becomes complex, *e.g.*, indiscernibility classes of data tables considered on higher level of the hierarchy can be equal to sets of paths of structural states. The theories of such data tables are much more complex than before. The rules in such a theory discovered from data may require extension to (spatio-)temporal decision rules or temporal association rules or even more complex rules defined by different temporal logics. The challenges are related to discovery of relevant rules and theories over such rules as well as to inducing, *e.g.*, Petri nets consistent with theories defined by such constraints.

11.9.7. Perception Based Computing

One of the main challenges in developing intelligent systems is discovering methods for approximate reasoning from measurements to perception, *i.e.*, deriving from concepts resulting from sensor measurements concepts to expressions enunciated in natural language that express perception understanding.

Nowadays, new emerging computing paradigms are investigated attempting to make progress in solving problems related to this challenge related to Perception Based Computing (PBC). Further progress depends on a successful cooperation of specialists from different scientific disciplines such as mathematics, computer science, artificial intelligence, biology, physics, chemistry, bioinformatics, medicine, neuroscience, linguistics, psychology, sociology. In particular, different aspects of reasoning from measurements to perception are investigated in psychology [10, 78], neuroscience [220], layered learning [294], mathematics of learning [220], machine learning, pattern recognition [81], data mining [104] and also by researchers working on recently emerged computing paradigms such as computing with words and perception [338], granular computing [194, 211], rough sets, rough-mereology, and rough-neural computing [194].

PBC provides capability to compute and reason with perception based information as humans do to perform a wide variety of physical and mental tasks without any measurement and computation. Reflecting the finite ability of the sensory organs and (finally the brain) to resolve details, perceptions are inherently granular. Boundaries of perceived granules (*e.g.*, classes) are unsharp and the values of the attributes they can take are granulated. In general, perception may be considered as understanding of sensory information. This point of view is, *e.g.*, presented in Computing with Words and Perception [338] which

derives from the fact that it opens the door to computation and reasoning with information which is perception – rather than measurement-based. Perceptions play a key role in human cognition, and underlie the remarkable human capability to perform a wide variety of physical and mental tasks without any measurements and any computations. Everyday examples of such tasks are driving a car in city traffic, playing tennis and summarizing a story.

The need for PBC appears, *e.g.*, in problems of analysis of complex processes that result from the interaction of many component processes and from control over such process. A component process control is aimed at achieving the desired patterns of the system behaviors. This task is a challenge for areas such as multi-agent systems or autonomous systems. Perceived properties of complex processes are often complex vague concepts, about which only partial information is available. Also information about the satisfiability of such concepts determines activating complex actions. It is worth noting that actions can be initiated at different levels of the hierarchy of concepts from a given ontology and that a prediction of action activation at higher levels of the hierarchy is usually conditioned by the perception of properties depending on the history of computations in which information about actions conducted also on lower levels of the hierarchy and their results is stored. Discovery of search strategies for new essential properties at higher levels of hierarchy becomes a challenge and is crucial for understanding of perception. The values of these attributes depend on the history of computing (with registered information about the actions performed on the actual and the lower levels and their results). These new features determine the perception of satisfiability degrees of complex concepts mentioned above conditioning execution of actions on the considered level of hierarchy. The difficulties of analysis and synthesis of perceptual computations follow from the nature of interactive compu-

tations, in which it becomes necessary to take into account interactions between processes during performed steps of computing (called intrastep interactions [77]). These difficulties follow also from partial information about component processes, from possible interactions between them, and also from requirements on avoidance of central control.

There are several critical issues for making progress in perception understanding and modeling. Among them in the nature of objects on which are performed computations leading to perception. We proposed to use granules for modeling such objects. The computations on granules are realized through interactions.

Note also that the fusion of information may lead to new information systems with structural objects [275–277] or to nets of information systems linked by different constraints. For example, a family of parameterized sensors may model a situation in which the sensors are enabled by the judgment module for recording features of video at different moments of time in probing the environment. This makes it possible to collect the necessary features of the environment for an activating of the relevant higher level action. Parameters may be related, *e.g.*, to positions of moving camera. This is closely related to the approach to perception presented in the book by Noë [177] (page 1):

[...] perceiving is a way of acting. Perception is not something that happens to us, or in us. It is something we do. Think of blind person tap-tapping his or her way around a cluttered space, perceiving the space by touch, not all at once, but through time, by skillful probing and movement. This is, or at least ought to be, our paradigm of what perceiving is. The world makes itself available to the perceiver through physical movement and interaction.

The developed methods for PBC are using the generalized information systems (a special kind of data tables) and the rough set approach for representing partial information on interactions in layered granular networks [92, 93, 276, 277]. The idea of the representation of interactions using information systems has some roots in such approaches as rough sets introduced by Zdzisław Pawlak [203], the information flow by Jon Barwise [11] or Chu spaces (<http://chu.stanford.edu/>). Information systems are used to represent granules of different complexity and the interactions among them [93, 276]. Rough sets are used for vague concept approximation [210], *e.g.*, in the approximation of ontologies given by experts [12].

All the above considered challenges lead us to conclusion that our model of computation on granules should consists as the fundamental concept the concept of interaction among granules. Recently the approach to PBC was extended to interactive granular computations (see, *e.g.*, [91, 95, 275]). This issue is shortly discussed in the following section.

11.10. Rough Sets and Logic

Rough set theory has contributed to some extent to various kinds of deductive reasoning. Particularly, various kinds of logics based on the rough set approach have been investigated, rough set methodology contributed essentially to modal logics, many valued logic, intuitionistic logic and others. A summary of this research can be found in the books [29, 38, 181, 221] and interested reader is advised to consult these volumes.

The creation of computers and their innovative applications essentially contributed to the rapid growth of interest in inductive reasoning [46, 139]. This domain develops very dynamically thanks to computer science. Machine learning, knowledge discovery, reasoning from data, expert systems and others are examples of new directions in inductive reasoning. It seems that rough set theory is very well suited as a theoretical basis for inductive reasoning. Basic concepts of this theory fit very well to represent and analyze knowledge acquired from examples, which can be next used as starting point for generalization. Besides, in fact rough set theory has been successfully applied in many domains to find patterns in data (data mining) and acquire knowledge from examples (learning from examples). Thus, rough set theory seems to be another candidate as a mathematical foundation of inductive reasoning [20, 172, 274].

The most interesting from computer science point of view is *common sense* reasoning. We use this kind of reasoning in our everyday life, and examples of such kind of reasoning we face in news papers, radio TV etc., in political, economic etc., debates and discussions.

The starting point to such reasoning is the knowledge possessed by the specific group of people (*common knowledge*) concerning some subject and intuitive methods of deriving conclusions from it. We do not have here possibilities of resolving the dispute by means of methods given by deductive logic (reasoning) or by inductive logic (experiment). So the best known methods of solving the dilemma is voting, negotiations or even war.

These methods do not reveal the truth or falsity of the thesis under consideration at all. Of course, such methods are not acceptable in math-

ematics or physics. Nobody is going to solve by voting, negotiations or declare a war – the truth of Fermat's theorem or Newton's laws.

Reasoning of this kind is the least studied from the theoretical point of view and its structure is not sufficiently understood, in spite of many interesting theoretical research in this domain [47]. The meaning of common sense reasoning, considering its scope and significance for some domains, is fundamental and rough set theory can also play an important role in it but more fundamental research must be done to this end [259].

There are numerous issues related to approximate reasoning under uncertainty. These issues are discussed in books on granular computing, rough mereology, and computational complexity of algorithmic problems related to these issues. For more detail, the reader is referred to the following books [37, 154, 157, 211, 222].

It is worthwhile to mention that still much more work should be done to develop approximate reasoning methods for making progress in development of intelligent systems. This idea was very well expressed by Professor Leslie Valiant (see: people.seas.harvard.edu/~valiant/researchinterests.htm):

A fundamental question for artificial intelligence is to characterize the computational building blocks that are necessary for cognition. A specific challenge is to build on the success of machine learning so as to cover broader issues in intelligence.... This requires, in particular a reconciliation between two contradictory characteristics – the apparent logical nature of reasoning and the statistical nature of learning.

One of the challenges for intelligent systems based on I(R)GrC is to develop methods for reasoning over interactive computations performed on c-granules making it possible for agents to control such computations toward achieving the target tasks. Such reasoning we call *adaptive judgment* [91]. *Intuitive judgment* and *rational judgment* are distinguished as different kinds of judgment [100]. Adaptive judgment is the basic tool in discovering of relevant patterns of different complexity used for approximation of complex vague concepts and inducing approximate reasoning schemes on such approximations. Adaptive judgement in intelligent systems is a mixture of reasoning based on deduction, abduction, induction, case based or analogy based reasoning, experience, observed changes in the environment, meta-heuristics from natural computing. Let us also note the following remark [310]:

Practical judgment is not algebraic calculation. Prior to any deductive or inductive reckoning, the judge is involved in selecting objects and relationships for attention and assessing their interactions. Identifying things of importance from a potentially endless pool of candidates, assessing their relative significance, and evaluating their relationships is well beyond the jurisdiction of reason.

We would like to stress that still much more work should be done to develop approximate reasoning methods about complex vague concepts for making progress in development of intelligent systems, in particular for the risk management and cost/benefit analysis in such systems.

The question arises about the logic relevant for the above discussed tasks. Let us observe that the satisfiability relations in the I(R)GrC framework can be treated as tools for constructing new information granules. If fact, for a given satisfiability relation, the semantics of formulas relative to this relation is defined. In this way the candidates for new relevant information granules are obtained. We would like to emphasize a very important feature. The relevant satisfiability relation for the considered problems is not given but it should be induced (discovered) on the basis of a partial information encoded in information (decision) systems. For real-life problems, it is often necessary to discover a hierarchy of satisfiability relations before we obtain the relevant target level. Information granules constructed at different levels of this hierarchy finally lead to relevant ones for approximation of complex vague concepts related to complex information granules expressed in natural language (see Figure 11.2).

Let us consider one more example showing the adaptive judgment importance. One of the main challenges in intelligent systems is to discover of the high quality approximations of vague complex concepts initiating actions. This task may be considered as the complex game discovery task (see Figure 11.3) from data and domain knowledge. The agents are using the discovered games for achieving their targets in the environment. The discovery process can be based on hierarchical learning supported by domain knowledge [12, 91]. Such games are evolving in time (drifting in time) together with data and knowledge about the approximated concepts. The relevant adaptive strategies for adapting the games to changes perceived by agents are required. These adaptive strategies are used to control the behavior of agents toward achieving by them targets. Note that also these strategies should be learned from data and domain knowledge. Approximations of the mentioned above complex vague concepts are based on complex patterns

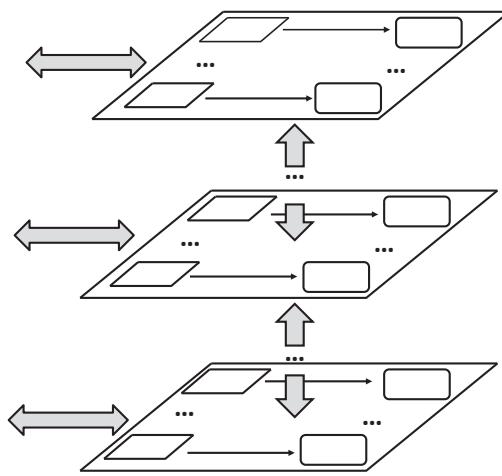


Fig. 11.2. Interactive hierarchical structures (gray arrows show interactions between hierarchical levels and the environment, arrows at hierarchical levels point from information (decision) systems representing partial specifications of satisfiability relations to induced from them theories consisting of rule sets).

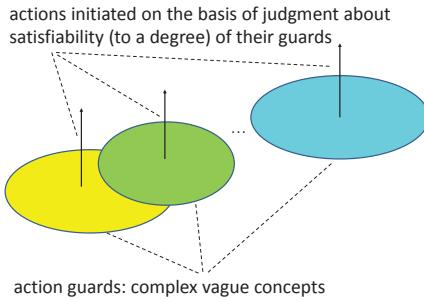


Fig. 11.3. Games based on complex vague concepts.

(or computational building blocks, using the Valiant words). Searching for and/or discovering of such patterns is based on adaptive judgment.

11.11. Conclusions

In the chapter, rudiments on rough sets are presented as well as the rough set based methods in pattern recognition are outlined. We have also listed some research directions based on I(R)GrC which are important

for discovery of complex patterns and/or computational building blocks for cognition.

Acknowledgments

This work was partially supported by the Polish National Science Centre (NCN) grants DEC-2011/01/D /ST6/06981, DEC-2013/09/B/ST6/01568 as well as by the Polish National Centre for Research and Development (NCBiR).

References

- [1] Albanese, A., Sankar K. Pal, F. and Petrosino, A. (2014). Rough sets, kernel set, and spatiotemporal outlier detection, *IEEE Transactions on Knowledge and Data Engineering* **26**, pp. 194–207.
- [2] An, A., Huang, Y., Huang, X. and Cercone, N. (2004). Feature selection with rough sets for web page classification, in [216], pp. 1–13.
- [3] An, S., Shi, H., Hu, Q., Li, X. and Dang, J. (2014). Fuzzy rough regression with application to wind speed prediction, *Information Sciences* **282**, pp. 388–400.
- [4] Baker, G. P. and Hacker, P. M. S. (eds.) (2005). *Wittgenstein: Understanding and Meaning: Volume 1 of an Analytical Commentary on the Philosophical Investigations, Part II: Exegesis 1-184* (Blackwell), (2nd Edition).
- [5] Banerjee, M., Mitra, S. and Banka, H. (2007). Evolutionary-rough feature selection in gene expression data, *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* **37**, pp. 622–632.
- [6] Banerjee, M., Mitra, S. and Pal, S. K. (1998). Rough-fuzzy MLP, *IEEE Transactions on Neural Nets* **9**, pp. 1203–1216.
- [7] Banerjee, M. and Pal, S. K. (1996). Roughness of a fuzzy set, *Information Sciences* **93**, 3-4, pp. 235–246.
- [8] Banka, H. and Mitra, S. (2012). Feature selection, classification and rule generation using rough sets, in [214], pp. 51–76.
- [9] Bargiela, A. and Pedrycz, W. (eds.) (2003). *Granular Computing: An Introduction* (Kluwer Academic Publishers).
- [10] Barsalou, L. W. (1999). Perceptual symbol systems, *Behavioral and Brain Sciences* **22**, pp. 577–660.
- [11] Barwise, J. and Seligman, J. (1997). *Information Flow: The Logic of Distributed Systems* (Cambridge University Press).
- [12] Bazan, J. (2008a). Hierarchical classifiers for complex spatio-temporal concepts, *Transactions on Rough Sets IX: Journal Subline*, LNCS 5390, pp. 474–750.
- [13] Bazan, J. (2008b). Rough sets and granular computing in behavioral pattern identification and planning, in [211], pp. 777–822.

- [14] Bazan, J., Latkowski, R. and Szczyka, M. (2005a). DIXER - Distributed executor for Rough Set Exploration System, in *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2005), Regina, Canada, August 31-September 3, 2005, Part II*, pp. 362–371.
- [15] Bazan, J., Nguyen, H. S., Nguyen, S. H., Synak, P. and Wróblewski, J. (2000). Rough set algorithms in classification problems, in [223], pp. 49–88.
- [16] Bazan, J. and Skowron, A. (2005a). On-line elimination of non-relevant parts of complex objects in behavioral pattern identification, in [187], pp. 720–725.
- [17] Bazan, J., Szczyka, M., Wojna, M. and Wojnarski, M. (2004). On the evolution of Rough Set Exploration System, in [313], pp. 592–601.
- [18] Bazan, J. G. (1998). A comparison of dynamic and non-dynamic rough set methods for extracting laws from decision tables, in [226], pp. 321–365.
- [19] Bazan, J. G., Nguyen, H. S., Peters, J. F., Skowron, A. and Szczyka, M. (2003). Rough set approach to pattern extraction from classifiers, in *Proceedings of the Workshop on Rough Sets in Knowledge Discovery and Soft Computing at ETAPS 2003, April 12-13, 2003*, pp. 20–29.
- [20] Bazan, J. G., Peters, J. F. and Skowron, A. (2005b). Behavioral pattern identification through rough set modelling, in *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2005), Regina, Canada, August 31-September 3, 2005, Part II*, pp. 688–697.
- [21] Bazan, J. G. and Skowron, A. (2005b). Classifiers based on approximate reasoning schemes, in B. Dunin-Keplicz, A. Jankowski, A. Skowron and M. Szczyka (eds.), *Monitoring, Security, and Rescue Tasks in Multiagent Systems (MSRAS'2004)*, Advances in Soft Computing (Springer, Heidelberg), pp. 191–202.
- [22] Bazan, J. G. and Szczyka, M. (2001). RSES and RSESlib - a collection of tools for rough set computations, in [345], pp. 106–113.
- [23] Bello, R., Falcón, R. and Pedrycz, W. (2010). *Granular Computing: At the Junction of Rough Sets and Fuzzy Sets, Studies in Fuzziness and Soft Computing*, Vol. 234 (Springer, Heidelberg).
- [24] Borrett, S. R., Bridewell, W., Langley, P. and Arrigo, K. R. (2007). A method for representing and developing process models, *Ecological Complexity* **4**, pp. 1–12.
- [25] Bower, J. M. and Bolouri, H. (eds.) (2001). *Computational Modeling of Genetic and Biochemical Networks* (MIT Press).
- [26] Breiman, L. (2001). Statistical modeling: The two cultures, *Statistical Science* **16**, 3, pp. 199–231.
- [27] Brown, F. (1990). *Boolean Reasoning* (Kluwer Academic Publishers, Dordrecht).
- [28] Bums, L. C. (1991). *Vagueness. An Investigation into Natural Languages and the Sorites Paradox* (Kluwer).

- [29] Chakraborty, M. and Pagliani, P. (2008). *A Geometry of Approximation: Rough Set Theory: Logic, Algebra and Topology of Conceptual Patterns* (Springer, Heidelberg).
- [30] Chikalov, I., Lozin, V., Lozina, I., Moshkov, M., H. S. Nguyen, Skowron, A. and Zielosko, B. (2012). *Three Approaches to Data Analysis. Test Theory, Rough Sets and Logical Analysis of Data, Series Intelligent Systems Reference Library*, Vol. 41 (Springer).
- [31] Chmielewski, M. R. and Grzymała-Busse, J. W. (1996). Global discretization of continuous attributes as preprocessing for machine learning, *International Journal of Approximate Reasoning* **15**, 4, pp. 319–331.
- [32] Choubey, S. K., Deogun, J. S., Raghavan, V. V. and Sever, H. (1996). A comparison of feature selection algorithms in the context of rough classifiers, in F. Petry (ed.), *International Conference on Fuzzy Systems (FUZZ-IEEE '1996), September 8-11, 1996, New Orleans, LA*, Vol. 2 (IEEE Service Center, Piscataway, NJ), pp. 1122–1128.
- [33] Cios, K., Pedrycz, W., Swiniarski, R. and Kurgan, L. A. (2007). *Data mining. A Knowledge Discovery Approach* (Springer, New York).
- [34] Cornelis, C., Jensen, R., Martí, G. H. and Ślezak, D. (2010a). Attribute selection with fuzzy decision reducts, *Information Sciences* **180**, 2, pp. 209–224.
- [35] Cornelis, C., Jensen, R., Martín, G. H. and Ślezak, D. (2010b). Stable rule extraction and decision making in rough non-deterministic information analysis, *Information Sciences* **180**, 2, pp. 209–224.
- [36] de Medeiros, A. K. A., Weijters, A. J. M. M. and van der Aalst, W. M. P. (2007). Genetic process mining: An experimental evaluation, *Data Mining and Knowledge Discovery* **14**, pp. 245–304.
- [37] Delimata, P., Moshkov, M. J., Skowron, A. and Suraj, Z. (2009). *Inhibitory Rules in Data Analysis: A Rough Set Approach, Studies in Computational Intelligence*, Vol. 163 (Springer, Heidelberg).
- [38] Demri, S. and Orłowska, E. (eds.) (2002). *Incomplete Information: Structure, Inference, Complexity*, Monographs in Theoretical Computer Science (Springer-Verlag, Heidelberg).
- [39] Deogun, J., Raghavan, V. V., Sarkar, A. and Sever, H. (1997). Data mining: Trends in research and development, in [121], pp. 9–46.
- [40] Dubois, V. and Quafafou, M. (2002). Concept learning with approximation: Rough version spaces, in *Third International Conference on Rough Sets and Current Trends in Computing (RSCTC'2002), Malvern, PA, October 14-16, 2002*, pp. 239–246.
- [41] Duda, R., Hart, P. and Stork, R. (2002). *Pattern Classification* (John Wiley & Sons, New York, NY).
- [42] Düntsch, I. and Gediga, G. (2000). *Rough set data analysis: A road to non-invasive knowledge discovery* (Methodos Publishers, Bangor, UK).
- [43] Fahle, M. and Poggio, T. (2002). *Perceptual Learning* (MIT Press, Cambridge).
- [44] Feng, J., Jost, J. and Minping, Q. (eds.) (2007). *Network: From Biology to Theory* (Springer, Berlin).

- [45] Frege, G. (1903). *Grundgesetzen der Arithmetik, 2* (Verlag von Hermann Pohle, Jena).
- [46] Gabbay, D. M., Hartmann, S. and Woods, J. (eds.) (2011). *Inductive Logic*, Handbook of the History of Logic (Elsevier, North-Holland, Oxford, UK).
- [47] Gabbay, D. M., Hogger, C. J. and Robinson, J. A. (eds.) (1994). *Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 3: Non-monotonic Reasoning and Uncertain Reasoning* (Calderon Press, Oxford).
- [48] Ganivada, A., Dutta, S. and Pal, S. (2011). Fuzzy rough granular neural networks, Fuzzy granules and classification, *Theoretical Computer Science* **412**, 42, pp. 5834–5853.
- [49] Ganivada, A. and Pal, S. K. (2011). A novel fuzzy rough granular neural network for classification, *International Journal of Computational Intelligence Systems* **4**, pp. 1042–1051.
- [50] Ganivada, A., Ray, S. and Pal, S. (2013). Fuzzy rough sets, and a granular neural network for unsupervised feature selection, *Neural Networks* **48**, pp. 91–108.
- [51] Ganivada, A., Ray, S. S. and Pal, S. (2012). Fuzzy rough granular self-organizing map and fuzzy rough entropy, *Theoretical Computer Science* **466**, pp. 37–63.
- [52] Garcia-Molina, H., Ullman, J. and Widom, J. (2002). *Database Systems: The Complete Book* (Prentice Hall, Upper Saddle River, New Jersey).
- [53] Geach, P. and Black, M. (eds.) (1960). *Translations from the Philosophical Writings of Gottlob Frege* (Blackwell).
- [54] Gediga, G. and Düntschi, I. (2001). Rough approximation quality revisited, *Artificial Intelligence* **132**, pp. 219–234.
- [55] Gediga, G. and Düntschi, I. (2004). On model evaluation, indices of importance, and interaction values in rough set analysis, in [194], pp. 251–276.
- [56] Gell-Mann, M. (1994). *The Quark and the Jaguar - Adventures in the Simple and the Complex* (Brown and Co., London).
- [57] Goldin, D., Smolka, S. and Wegner, P. (eds.) (2006). *Interactive Computation: The New Paradigm* (Springer).
- [58] Góra, G. and Wojna, A. G. (2002). RIONA: A new classification system combining rule induction and instance-based learning, *Fundamenta Informaticae* **51**, 4, pp. 369–390.
- [59] Greco, S., Inuiguchi, M. and Słowiński, R. (2006). Fuzzy rough sets and multiple-premise gradual decision rules. *International Journal of Approximate Reasoning* **41**, 2, pp. 179–211.
- [60] Greco, S., Matarazzo, B. and Słowiński, R. (2000). Dealing with missing data in rough set analysis of multi-attribute and multi-criteria decision problems, in S. Zanakis, G. Doukidis and C. Zopounidis (eds.), *Decision Making: Recent Developments and Worldwide Applications* (Kluwer Academic Publishers, Boston, MA), pp. 295–316.
- [61] Greco, S., Matarazzo, B. and Słowiński, R. (2001). Rough set theory for multicriteria decision analysis, *European Journal of Operational Research* **129**, 1, pp. 1–47.

- [62] Greco, S., Matarazzo, B. and Słowiński, R. (2004a). Dominance-based rough set approach to knowledge discovery (I) - General perspective, (II) - Extensions and applications, in [341], pp. 513–552, 553–612.
- [63] Greco, S., Matarazzo, B. and Słowiński, R. (2004b). Dominance-based rough set approach to knowledge discovery (II) – Extensions and applications, in [341], pp. 553–612.
- [64] Greco, S., Matarazzo, B. and Słowiński, R. (2007). Dominance-based rough set approach as a proper way of handling graduality in rough set theory, *Transactions on Rough Sets VII: Journal Subline*, LNCS 4400, pp. 36–52.
- [65] Greco, S., Matarazzo, B. and Słowiński, R. (2009). Granular computing and data mining for ordered data: The dominance-based rough set approach, in *Encyclopedia of Complexity and Systems Science* (Springer, Heidelberg), pp. 4283–4305.
- [66] Grzymała-Busse, J. W. (1990). *Managing Uncertainty in Expert Systems* (Kluwer Academic Publishers, Norwell, MA).
- [67] Grzymała-Busse, J. W. (1992). LERS – A system for learning from examples based on rough sets, in [287], pp. 3–18.
- [68] Grzymała-Busse, J. W. (1997a). Classification of unseen examples under uncertainty, *Fundamenta Informaticae* **30**, 3-4, pp. 255–267.
- [69] Grzymała-Busse, J. W. (1997b). A new version of the rule induction system LERS, *Fundamenta Informaticae* **31**, 1, pp. 27–39.
- [70] Grzymała-Busse, J. W. (2004). Three strategies to rule induction from data with numerical attributes, in [216], pp. 54–62.
- [71] Grzymała-Busse, J. W. (2005a). LERS - A data mining system, in [129], pp. 1347–1351.
- [72] Grzymała-Busse, J. W. (2005b). Rule induction, in [129], pp. 277–294.
- [73] Grzymała-Busse, J. W. (2011). Generalized parameterized approximations, in *Proceedings of the 6th International Conference on Rough Sets and Knowledge Technology (RSKT'2011), Banff, Canada, October 9-12, 2011*, pp. 136–145.
- [74] Grzymała-Busse, J. W. (2015). Rule induction from rough approximations, in R. Słowiński and Y. Yao (eds.), *Handbook of Computational Intelligence, Part C* (Springer, Heidelberg), pp. 371–385.
- [75] Grzymała-Busse, J. W. and Grzymała-Busse, W. J. (2005). Handling missing attribute values, in [129], pp. 37–57.
- [76] Grzymała-Busse, J. W. and Ziarko, W. (2000). Data mining and rough set theory, *Communications of the ACM* **43**, pp. 108–109.
- [77] Gurevich, Y. (2006). Interactive algorithms 2005, in [57], pp. 165–181.
- [78] Harnad, S. (1987). *Categorical Perception: The Groundwork of Cognition* (Cambridge University Press, New York, NY).
- [79] Hassanien, A. E., Sakai, H., Ślezak, D., Chakraborty, M. K. and Zhu, W. (2011). Special issue: Rough and fuzzy methods for data mining (preface), *International Journal of Hybrid Intelligent Systems* **8**, 1, pp. 1–2.
- [80] Hassanien, A. E., Suraj, Z., Slezak, D. and Lingras, P. (eds.) (2008). *Rough Computing: Theories, Technologies and Applications* (IGI Global, Hershey, PA).

- [81] Hastie, T., Tibshirani, R. and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Springer-Verlag, Heidelberg).
- [82] Herbert, J. and Yao, J. T. (2005). Time-series data analysis with rough sets, in *Proceedings of the 4th International Conference on Computational Intelligence in Economics and Finance (CIEF'2005), Salt Lake City, UT, July 21-26, 2005*, pp. 908–911.
- [83] Hu, Q., Pedrycz, W., Yu, D. and Lang, J. (2010). Selecting discrete and continuous features based on neighborhood decision error minimization, *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics* **40**, pp. 137–150.
- [84] Hu, Q., Yu, D., Pedrycz, W. and Chen, D. (2011). Kernelized fuzzy rough sets and their applications, *IEEE Transactions on Knowledge and Data Engineering* **23**, pp. 1649–1667.
- [85] Hu, Q. H., Yu, D. R., Liu, J. F. and X.Wu, C. (2008). Neighborhood rough set based heterogeneous feature subset selection, *Information Sciences* **178**, pp. 3577–3594.
- [86] Hu, X. and Cercone, N. (1995). Learning in relational databases: A rough set approach, *Computational Intelligence: An International Journal* **11**, 2, pp. 323–338.
- [87] Hu, X. and Cercone, N. (1999). Data mining via discretization, generalization and rough set feature selection, *Knowledge and Information Systems: An International Journal* **1**, 1, pp. 33–60.
- [88] Hu, X. and Cercone, N. (2001). Discovering maximal generalized decision rules through horizontal and vertical data reduction, *Computational Intelligence: An International Journal* **17**, 4, pp. 685–702.
- [89] Huhns, M. N. and Singh, M. P. (1998). *Readings in Agents* (Morgan Kaufmann, San Mateo).
- [90] Inuiguchi, M., Wu, W.-Z., Cornelis, C. and Verbiest, N. (2015). Fuzzy-rough hybridization, in R. Słowiński and Y. Yao (eds.), *Handbook of Computational Intelligence, Part C* (Springer, Heidelberg), pp. 425–451.
- [91] Jankowski, A. (2016). Complex Systems Engineering: Wisdom for Saving Billions based on Interactive Granular Computing. (*in preparation*) (Springer, Heidelberg).
- [92] Jankowski, A. and Skowron, A. (2007). A Wistech paradigm for intelligent systems, *Transactions on Rough Sets VI: Journal Subline*, LNCS 4374, pp. 94–132.
- [93] Jankowski, A. and Skowron, A. (2008). Logic for artificial intelligence: The Rasiowa - Pawlak school perspective, in A. Ehrenfeucht, V. Marek and M. Srebrny (eds.), *Andrzej Mostowski and Foundational Studies* (IOS Press, Amsterdam), pp. 106–143.
- [94] Jankowski, A., Skowron, A. and Swiniarski, R. W. (2013). Interactive computations: Toward risk management in interactive intelligent systems, in P. Maji, A. Ghosh, M. N. Murty, K. Ghosh and S. K. Pal (eds.), *Pattern Recognition and Machine Intelligence - 5th International Conference, PReMI 2013, Kolkata, India, December 10-14, 2013. Proceedings, Lecture Notes in Computer Science*, Vol. 8251 (Springer), pp. 1–12.

- [95] Jankowski, A., Skowron, A. and Swiniarski, R. W. (2014). Interactive complex granules, *Fundamenta Informaticae* **133**, 2-3, pp. 181–196.
- [96] Janusz, A. and Ślezak, D. (2014). Rough set methods for attribute clustering and selection, *Applied Artificial Intelligence* **28**, 3, pp. 220–242.
- [97] Jensen, R. (2008). Rough set-based feature selection: A review, in A. E. Hassanien, Z. Suraj, D. Slezak and P. Lingras (eds.), *Rough Computing: Theories, Technologies and Applications* (IGI Global, Hershey, PA).
- [98] Jensen, R. and Shen, Q. (2008). *Computational Intelligence and Feature Selection: Rough and Fuzzy Approaches*, IEEE Press Series on Computationa Intelligence (IEEE Press and John Wiley & Sons, Hoboken, NJ).
- [99] Joshi, M., Lingras, P. and Rao, C. R. (2012). Correlating fuzzy and rough clustering, *Fundamenta Informaticae* **115**, 2-3, pp. 233–246.
- [100] Kahneman, D. (2002). Maps of bounded rationality: Psychology for behavioral economics, *The American Economic Review* **93**, pp. 1449–1475.
- [101] Keefe, R. (2000). *Theories of Vagueness* (Cambridge Studies in Philosophy, Cambridge, UK).
- [102] Kim, D. (2001). Data classification based on tolerant rough set, *Pattern Recognition* **34**, 8, pp. 1613–1624.
- [103] Kim, D. and Bang, S. Y. (2000). A handwritten numeral character classification using tolerant rough set, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **22**, 9, pp. 923–937.
- [104] Kloesgen, W. and Żytkow, J. (eds.) (2002). *Handbook of Knowledge Discovery and Data Mining* (Oxford University Press, Oxford).
- [105] Komorowski, J., Øhrn, A. and Skowron, A. (2000). Rosetta and other software systems for rough sets, in W. Klösgen and J. Żytkow (eds.), *Handbook of Data Mining and Knowledge Discovery* (Oxford University Press), pp. 554–559.
- [106] Kopczynski, M., Grzes, T. and Stepaniuk, J. (2014). FPGA in rough-granular computing: Reduct generation, in *2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), Warsaw, Poland, August 11-14, 2014 - Volume I* (IEEE Computer Society), pp. 364–370.
- [107] Kruczyk, M., Baltzer, N., Mieczkowski, J., Draminski, M., Koronacki, J. and Komorowski, J. (2013). Random reducts: A monte carlo rough set-based method for feature selection in large datasets, *Fundamenta Informaticae* **127**, 1-4, pp. 273–288.
- [108] Kryszkiewicz, M. and Cichoń, K. (2004). Towards scalable algorithms for discovering rough set reducts, in [216], pp. 120–143.
- [109] Kryszkiewicz, M. and Rybiński, H. (1996). Computation of reducts of composed information systems, *Fundamenta Informaticae* **27**, 2-3, pp. 183–195.
- [110] Latkowski, R. (2003). On decomposition for incomplete data, *Fundamenta Informaticae* **54**, 1, pp. 1–16.
- [111] Latkowski, R. (2005). Flexible indiscernibility relations for missing attribute values, *Fundamenta Informaticae* **67**, 1-3, pp. 131–147.
- [112] Leibniz, G. W. (1686). Discourse on metaphysics, in *Leibniz, G. W., Philosophical Essays*, pp. 35–68.

- [113] Leśniewski, S. (1929). Grungzüge eines neuen Systems der Grundlagen der Mathematik, *Fundamenta Mathematicae* **14**, pp. 1–81.
- [114] Leśniewski, S. (1982). On the foundations of mathematics, *Topoi* **2**, pp. 7–52.
- [115] Li, J. and Cercone, N. (2005). A rough set based model to rank the importance of association rules, in *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2005), Regina, Canada, August 31-September 3, 2005, Part II*, pp. 109–118.
- [116] Li, J. and Cercone, N. (2010). A method of discovering important rules using rules as attributes, *International Journal of Intelligent Systems* **25**, 2, pp. 180–206.
- [117] Li, X. and Rao, F. (2012). An rough entropy based approach to outlier detection, *Journal of Computational Information Systems* **8**, pp. 10501–10508.
- [118] Li, Y., Shiu, S. C.-K., Pal, S. K. and Liu, J. N.-K. (2006). A rough set-based case-based reasoner for text categorization, *International Journal of Approximate Reasoning* **41**, 2, pp. 229–255.
- [119] Lin, T. Y. (1989). Neighborhood systems and approximation in database and knowledge base systems, in M. L. Emrich, M. S. Phifer, M. Hadzikadic and Z. W. Ras (eds.), *Proceedings of the Fourth International Symposium on Methodologies of Intelligent Systems (Poster Session), October 12-15, 1989* (Oak Ridge National Laboratory, Charlotte, NC), pp. 75–86.
- [120] Lin, T. Y. (1998). The discovery, analysis and representation of data dependencies in databases, in L. Polkowski and A. Skowron (eds.), *Rough Sets in Knowledge Discovery 1: Methodology and Applications, Studies in Fuzziness and Soft Computing*, Vol. 18 (Physica-Verlag, Heidelberg), pp. 107–121.
- [121] Lin, T. Y. and Cercone, N. (eds.) (1997). *Rough Sets and Data Mining - Analysis of Imperfect Data* (Kluwer Academic Publishers, Boston, MA).
- [122] Lingras, P. (2001a). Fuzzy - rough and rough - fuzzy serial combinations in neurocomputing, *Neurocomputing* **36**, 1-4, pp. 29–44.
- [123] Lingras, P. (2001b). Unsupervised rough set classification using gas, *Journal of Intelligent Information Systems* **16**, 3, pp. 215–228.
- [124] Lingras, P. and Peters, G. (2011). Rough clustering, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* **1**, 1, pp. 64–72.
- [125] Lingras, P. and Peters, G. (2012). Applying rough set concepts to clustering, in [214], pp. 23–37.
- [126] Lingras, P. and West, C. J. (2004). Rough c-means, *Intelligent Information Systems* **23**, pp. 5–16.
- [127] Liu, J., Hu, Q. and Yu, D. (2008). A comparative study on rough set based class imbalance learning, *Knowledge-Based Systems* **21**, pp. 753–763.
- [128] Lukasiewicz, J. (1913). Die logischen Grundlagen der Wahrscheinlichkeitssrechnung, 1913, in L. Borkowski (ed.), *Jan Lukasiewicz - Selected Works* (North Holland Publishing Company, Amsterdam, London, Polish Scientific Publishers, Warsaw, 1970, pp. 16-63).

- [129] Maimon, O. and Rokach, L. (eds.) (2005). *The Data Mining and Knowledge Discovery Handbook* (Springer, Heidelberg).
- [130] Maji, P. and Pal, S. (2007a). RFCM: A hybrid clustering algorithm using rough and fuzzy sets, *Fundamenta Informaticae* **80**, 4, pp. 477–498.
- [131] Maji, P. and Pal, S. (2007b). Rough set based generalized fuzzy c-means algorithm and quantitative indices, *IEEE Transactions on Systems, Man and Cybernetics. Part B: Cybernetics* **37**, 6, pp. 1529–1540.
- [132] Maji, P. and Pal, S. (2010). Fuzzy-rough sets for information measures and selection of relevant genes from microarray data, *IEEE Transactions on Systems, Man and Cybernetics. Part B: Cybernetics* **40**, 3, pp. 741–752.
- [133] Maji, P. and Pal, S. K. (2012). *Rough-Fuzzy Pattern Recognition: Application in Bioinformatics and Medical Imaging*, Wiley Series in Bioinformatics (John Wiley & Sons, Hoboken, NJ).
- [134] Maji, P. and Paul, S. (2014). *Scalable Pattern Recognition Algorithms. Applications in Computational Biology and Bioinformatics* (Springer, Berlin).
- [135] Małyszko, D. and Stepaniuk, J. (2010). Adaptive multilevel rough entropy evolutionary thresholding, *Information Sciences* **180**, pp. 1138–1158.
- [136] Marcus, S. (1998). The paradox of the heap of grains, in respect to roughness, fuzziness and negligibility, in [225], pp. 19–23.
- [137] Mazumdar, D., Mitra, S. and Mitra, S. (2010). Evolutionary-rough feature selection for face recognition, *Transactions on Rough Sets IV: Journal Sub-line*, LNCS 6190, pp. 117–142.
- [138] Mehera, S. K. and Pal, S. K. (2011). Rough-wavelet granular space and classification of multispectral remote sensing image, *Applied Soft Computing* **11**, pp. 5662–5673.
- [139] Mill, J. S. (1862). *Ratiocinative and Inductive, Being a Connected View of the Principles of Evidence, and the Methods of Scientific Investigation* (Parker, Son, and Bourn, West Strand London).
- [140] Min, F., Hu, Q. and Zhu, W. (2014). Feature selection with test cost constraint. pp. 167–179.
- [141] Mitchel, T. M. (1999). *Machine Learning* (McGraw-Hill Series in Computer Science, Boston, MA).
- [142] Mitra, P., Mitra, S. and Pal, S. K. (1999). Modular rough fuzzy MLP: Evolutionary design, in *Proceedings of the 7-th International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing (RSFDGrC'99)*, Yamaguchi, November 9–11, 1999, pp. 128–136.
- [143] Mitra, P., Mitra, S. and Pal, S. K. (2001). Evolutionary modular design of rough knowledge-based network using fuzzy attributes, *Neurocomputing* **36**, pp. 45–66.
- [144] Mitra, P., Pal, S. and Siddiqi, M. A. (2003a). Nonconvex clustering using expectation maximization algorithm with rough set initialization, *Pattern Recognition Letters* **24**, pp. 863–873.
- [145] Mitra, P., Pal, S. K. and Siddiqi, M. A. (2003b). Non-convex clustering using expectation maximization algorithm with rough set initialization, *Pattern Recognition Letters* **24**, 6, pp. 863–873.
- [146] Mitra, S. (2004). An evolutionary rough partitive clustering, *Pattern Recognition Letters* **25**, pp. 1439–1449.

- [147] Mitra, S. and Acharya, T. (2003). *Data mining. Multimedia, Soft Computing, and Bioinformatics* (John Wiley & Sons, New York, NY).
- [148] Mitra, S. and Banka, H. (2007). Application of rough sets in pattern recognition, *Transactions on Rough Sets VII: Journal Subline*, LNCS 4400, pp. 151–169.
- [149] Mitra, S., Banka, H. and Pedrycz, W. (2006a). Rough-fuzzy c-means, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* **36**, pp. 795–805.
- [150] Mitra, S., Banka, H. and Pedrycz, W. (2006b). Rough-fuzzy collaborative clustering, *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews* **36**, pp. 795–805.
- [151] Mitra, S., Pedrycz, W. and Barman, B. (2010). Shadowed c-means: Integrating fuzzy and rough clustering, *Pattern Recognition* **43**, pp. 1282–1291.
- [152] Moshkov, M., Skowron, A. and Suraj, Z. (2006). On testing membership to maximal consistent extensions of information systems, in *Proceedings of the 5th International Conference on Rough Sets and Current Trends in Computing (RSCTC'2006)*, Kobe, Japan, November 6–8, 2006, pp. 85–90.
- [153] Moshkov, M., Skowron, A. and Suraj, Z. (2008a). On irreducible descriptive sets of attributes for information systems, in *Proceedings of the 6th International Conference on Rough Sets and Current Trends in Computing (RSCTC'2008)*, Akron, OH, USA, October 23–25, 2008, pp. 21–30.
- [154] Moshkov, M. J., Piliszczuk, M. and Zielosko, B. (2008b). *Partial Covers, Reducts and Decision Rules in Rough Sets - Theory and Applications, Studies in Computational Intelligence*, Vol. 145 (Springer, Heidelberg).
- [155] Moshkov, M. J., Skowron, A. and Suraj, Z. (2007). On minimal rule sets for almost all binary information systems, *Fundamenta Informaticae* **80**, 1–3, pp. 247–258.
- [156] Moshkov, M. J., Skowron, A. and Suraj, Z. (2009). On minimal inhibitory rules for almost all k-valued information systems, *Fundamenta Informaticae* **93**, 1–3, pp. 261–272.
- [157] Moshkov, M. J. and Zielosko, B. (2011). *Combinatorial Machine Learning - A Rough Set Approach, Studies in Computational Intelligence*, Vol. 360 (Springer, Heidelberg).
- [158] Nakata, M. and Sakai, H. (2005). Rough sets handling missing values probabilistically interpreted, in *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2005)*, Regina, Canada, August 31–September 3, 2005, Part I, pp. 325–334.
- [159] Nguyen, H. S. (1997). *Discretization of Real Value Attributes, Boolean Reasoning Approach*, Ph.D. thesis, Warsaw University, Warsaw, Poland.
- [160] Nguyen, H. S. (1998). From optimal hyperplanes to optimal decision trees, *Fundamenta Informaticae* **34**, 1–2, pp. 145–174.
- [161] Nguyen, H. S. (1999). Efficient SQL-learning method for data mining in large data bases, in T. Dean (ed.), *Sixteenth International Joint Conference on Artificial Intelligence IJCAI* (Morgan-Kaufmann Publishers, Stockholm, Sweden), pp. 806–811.

- [162] Nguyen, H. S. (2001). On efficient handling of continuous attributes in large data bases, *Fundamenta Informaticae* **48**, 1, pp. 61–81.
- [163] Nguyen, H. S. (2006). Approximate boolean reasoning: Foundations and applications in data mining, *Transactions on Rough Sets V: Journal Subline*, LNCS 4100, pp. 344–523.
- [164] Nguyen, H. S., Jankowski, A., Skowron, A., Stepaniuk, J. and Szczuka, M. (2010). Discovery of process models from data and domain knowledge: A rough-granular approach, in J. T. Yao (ed.), *Novel Developments in Granular Computing: Applications for Advanced Human Reasoning and Soft Computation* (IGI Global, Hershey, PA), pp. 16–47.
- [165] Nguyen, H. S. and Nguyen, S. H. (1998). Pattern extraction from data, *Fundamenta Informaticae* **34**, pp. 129–144.
- [166] Nguyen, H. S. and Nguyen, S. H. (1999). Rough sets and association rule generation, *Fundamenta Informaticae* **40**, 4, pp. 383–405.
- [167] Nguyen, H. S. and Skowron, A. (1995). Quantization of real value attributes, in *Proceedings of the Second Joint Annual Conference on Information Sciences, Wrightsville Beach, North Carolina, 1995, USA*, pp. 34–37.
- [168] Nguyen, H. S. and Skowron, A. (2008). A rough granular computing in discovery of process models from data and domain knowledge, *Journal of Chongqing University of Post and Telecommunications* **20**, 3, pp. 341–347.
- [169] Nguyen, H. S. and Skowron, A. (2013). Rough sets: From rudiments to challenges, in [271], pp. 75–173.
- [170] Nguyen, H. S. and Ślęzak, D. (1999). Approximate reducts and association rules - correspondence and complexity results, in *Proceedings of the 7-th International Workshop on Rough Sets, Fuzzy Sets, Data Mining, and Granular-Soft Computing (RSFDGrC'1999), Yamaguchi, November 9-11, 1999*, pp. 137–145.
- [171] Nguyen, S. H. (2000). Regularity analysis and its applications in data mining, in [223], pp. 289–378.
- [172] Nguyen, S. H., Bazan, J., Skowron, A. and Nguyen, H. S. (2004). Layered learning for concept synthesis, in [215], pp. 187–208.
- [173] Nguyen, S. H. and Nguyen, H. S. (1996). Some efficient algorithms for rough set methods, in *Sixth International Conference on Information Processing and Management of Uncertainty on Knowledge Based Systems IPMU'1996*, Vol. III (Granada, Spain), pp. 1451–1456.
- [174] Nguyen, T. T. (2005). Eliciting domain knowledge in handwritten digit recognition, in [187], pp. 762–767.
- [175] Nguyen, T. T. and Skowron, A. (2003). Rough set approach to domain knowledge approximation, in *Proceedings of the 9-th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2003), Chongqing, China, May 26-29, 2003*, pp. 221–228.
- [176] Nguyen, T. T. and Skowron, A. (2009). Rough-granular computing in human-centric information processing, in A. Bargiela and W. Pedrycz (eds.), *Human-Centric Information Processing Through Granular Modelling, Studies in Computational Intelligence*, Vol. 182 (Springer, Heidelberg), pp. 1–30.

- [177] Noë, A. (2004). *Action in Perception* (MIT Press).
- [178] Omicini, A., Ricci, A. and Viroli, M. (2006). The multidisciplinary patterns of interaction from sciences to computer science, in [57], pp. 395–414.
- [179] Orłowska, E. (1984). Semantics of vague concepts, in G. Dorn and P. Weingartner (eds.), *Foundation of Logic and Linguistics* (Plenum Press, New York), pp. 465–482.
- [180] Orłowska, E. (1987). Reasoning about vague concepts, *Bulletin of the Polish Academy of Sciences, Mathematics* **35**, pp. 643–652.
- [181] Orłowska, E. (ed.) (1997). *Incomplete Information: Rough Set Analysis, Studies in Fuzziness and Soft Computing*, Vol. 13 (Springer-Verlag/Physica-Verlag, Heidelberg).
- [182] Pal, S. (2010). Computational theory of perception (CTP), rough-fuzzy uncertainty analysis and mining in bioinformatics and web intelligence: A unified framework, *Transactions on Rough Sets XI: Journal Subline*, LNCS 5946, pp. 106–129.
- [183] Pal, S., Meher, S. and Dutta, S. (2012). Class-dependent rough-fuzzy granular space, dispersion index and classification, *Pattern Recognition* **45**, pp. 2690–2707.
- [184] Pal, S. and Pal, A. (eds.) (2001). *Pattern Recognition. From Classical to Modern Approaches* (World Scientific).
- [185] Pal, S. and Shiu, S. (2004). *Foundations of Soft Case-Based Reasoning* (Wiley, Hoboken, NJ).
- [186] Pal, S. K. (2004). Soft data mining, computational theory of perceptions, and rough-fuzzy approach, *Information Sciences* **163**, 1-3, pp. 5–12.
- [187] Pal, S. K., Bandopadhyay, S. and Biswas, S. (eds.) (2005a). *Proceedings of the First International Conference on Pattern Recognition and Machine Intelligence (PReMI'05) December 18–22, 2005, Indian Statistical Institute, Kolkata, Lecture Notes in Computer Science*, Vol. 3776 (Springer, Heidelberg).
- [188] Pal, S. K., Dasgupta, B. and Mitra, P. (2004a). Rough self organizing map, *Applied Intelligence* **21**, pp. 289–299.
- [189] Pal, S. K. and Mitra, P. (2002). Multispectral image segmentation using the rough-set-initialized EM algorithm, *IEEE Transactions on Geoscience and Remote Sensing* **40**, pp. 2495–2501.
- [190] Pal, S. K. and Mitra, P. (2004a). Case generation using rough sets with fuzzy representation, *IEEE Trans. Knowledge and Data Engineering* **16**, 3, pp. 292–300.
- [191] Pal, S. K. and Mitra, P. (2004b). *Pattern Recognition Algorithms for Data Mining* (CRC Press, Boca Raton, Florida).
- [192] Pal, S. K., Pedrycz, W., Skowron, A. and Swiniarski, R. (eds.) (2001). *Special volume: Rough-neuro computing, Neurocomputing*, Vol. 36.
- [193] Pal, S. K. and Peters, J. F. (eds.) (2010). *Rough Fuzzy Image Analysis Foundations and Methodologies* (Chapman & Hall/CRC, Boca Raton, Fl).
- [194] Pal, S. K., Polkowski, L. and Skowron, A. (eds.) (2004b). *Rough-Neural Computing: Techniques for Computing with Words*, Cognitive Technologies (Springer-Verlag, Heidelberg).

- [195] Pal, S. K., Shankar, B. U. and Mitra, P. (2005b). Granular computing, rough entropy and object extraction, *Pattern Recognition Letters* **26**, pp. 2509–2517.
- [196] Pal, S. K. and Skowron, A. (eds.) (1999). *Rough Fuzzy Hybridization: A New Trend in Decision-Making* (Springer-Verlag, Singapore).
- [197] Pancerz, K. and Suraj, Z. (2003). Modelling concurrent systems specified by dynamic information systems: A rough set approach, *Electronic Notes in Theoretical Computer Science* **82**, 4, pp. 206–218.
- [198] Pancerz, K. and Suraj, Z. (2004). Discovering concurrent models from data tables with the ROSECON system, *Fundamenta Informaticae* **60**, 1-4, pp. 251–268.
- [199] Pancerz, K. and Suraj, Z. (2006). Reconstruction of concurrent system models described by decomposed data tables, *Fundamenta Informaticae* **71**, 1, pp. 121–137.
- [200] Pancerz, K. and Suraj, Z. (2007). Towards efficient computing consistent and partially consistent extensions of information systems, *Fundamenta Informaticae* **79**, 3-4, pp. 553–566.
- [201] Papageorgiou, E. I. and Stylios, C. D. (2008). Fuzzy cognitive maps, in [211], pp. 755–774.
- [202] Pawlak, Z. (1981). Information systems - theoretical foundations, *Information Systems* **6**, pp. 205–218.
- [203] Pawlak, Z. (1982). Rough sets, *International Journal of Computer and Information Sciences* **11**, pp. 341–356.
- [204] Pawlak, Z. (1991). *Rough Sets: Theoretical Aspects of Reasoning about Data, System Theory, Knowledge Engineering and Problem Solving*, Vol. 9 (Kluwer Academic Publishers, Dordrecht, The Netherlands), ISBN 0-7923-1472-7.
- [205] Pawlak, Z. (1992). Concurrent versus sequential - the rough sets perspective, *Bulletin of the EATCS* **48**, pp. 178–190.
- [206] Pawlak, Z., Polkowski, L. and Skowron, A. (2000). Rough sets and rough logic: A KDD perspective, in [223], pp. 583–646.
- [207] Pawlak, Z. and Skowron, A. (1994). Rough membership functions, in R. Yager, M. Fedrizzi and J. Kacprzyk (eds.), *Advances in the Dempster-Shafer Theory of Evidence* (John Wiley & Sons, New York, NY), pp. 251–271.
- [208] Pawlak, Z. and Skowron, A. (2007a). Rough sets and boolean reasoning, *Information Sciences* **177**, 1, pp. 41–73.
- [209] Pawlak, Z. and Skowron, A. (2007b). Rough sets: Some extensions, *Information Sciences* **177**, 28-40, p. 1.
- [210] Pawlak, Z. and Skowron, A. (2007c). Rudiments of rough sets, *Information Sciences* **177**, 1, pp. 3–27.
- [211] Pedrycz, W., Skowron, S. and Kreinovich, V. (eds.) (2008). *Handbook of Granular Computing* (John Wiley & Sons, Hoboken, NJ).
- [212] Peters, G. (2006). Some refinements of rough k-means clustering, *Pattern Recognition* **39**, pp. 1481–1491.

- [213] Peters, G., Crespo, F., Lingras, P. and Weber, R. (2013). Soft clustering - fuzzy and rough approaches and their extensions and derivatives, *International Journal of Approximate Reasoning* **54**, 2, pp. 307–322.
- [214] Peters, G., Lingras, P., Ślęzak, D. and Yao, Y. (eds.) (2012). *Rough Sets: Selected Methods and Applications in Management and Engineering* (Springer, London).
- [215] Peters, J. F. and Skowron, A. (eds.) (2004). *Transactions on Rough Sets I: Journal Subline*, LNCS 3100.
- [216] Peters, J. F., Skowron, A., Dubois, D., Grzymala-Busse, J. W., Inuiguchi, M. and Polkowski, L. (eds.) (2004). *Transactions on Rough Sets II. Rough sets and fuzzy sets: Journal Subline*, LNCS 3135.
- [217] Peters, J. F., Skowron, A. and Suraj, Z. (2000a). An application of rough set methods in control design, *Fundamenta Informaticae* **43**, 1-4, pp. 269–290.
- [218] Peters, J. F., Skowron, A. and Suraj, Z. (2000b). An application of rough set methods in control design, *Fundamenta Informaticae* **43**, 1-4, pp. 269–290.
- [219] Peters, J. F., Suraj, Z., Shan, S., Ramanna, S., Pedrycz, W. and Pizzi, N. J. (2003). Classification of meteorological volumetric radar data using rough set methods. *Pattern Recognition Letters* **24**, 6, pp. 911–920.
- [220] Poggio, T. and Smale, S. (2003). The mathematics of learning: Dealing with data, *Notices of the AMS* **50**, 5, pp. 537–544.
- [221] Polkowski, L. (2002). *Rough Sets: Mathematical Foundations*, Advances in Soft Computing (Physica-Verlag, Heidelberg).
- [222] Polkowski, L. (ed.) (2011). *Approximate Reasoning by Parts. An Introduction to Rough Mereology*, Intelligent Systems Reference Library, Vol. 20 (Springer, Heidelberg).
- [223] Polkowski, L., Lin, T. Y. and Tsumoto, S. (eds.) (2000). *Rough Set Methods and Applications: New Developments in Knowledge Discovery in Information Systems, Studies in Fuzziness and Soft Computing*, Vol. 56 (Springer-Verlag/Physica-Verlag, Heidelberg).
- [224] Polkowski, L. and Skowron, A. (1996). Rough mereology: A new paradigm for approximate reasoning, *International Journal of Approximate Reasoning* **15**, 4, pp. 333–365.
- [225] Polkowski, L. and Skowron, A. (eds.) (1998a). *Proceedings of the First International Conference on Rough Sets and Soft Computing (RSCTC 1998), June 22–26, 1998, Warsaw, Poland, Lecture Notes in Artificial Intelligence*, Vol. 1424 (Springer-Verlag).
- [226] Polkowski, L. and Skowron, A. (eds.) (1998b). *Rough Sets in Knowledge Discovery 1: Methodology and Applications, Studies in Fuzziness and Soft Computing*, Vol. 18 (Physica-Verlag, Heidelberg).
- [227] Polkowski, L. and Skowron, A. (eds.) (1998c). *Rough Sets in Knowledge Discovery 2: Applications, Case Studies and Software Systems, Studies in Fuzziness and Soft Computing*, Vol. 19 (Physica-Verlag, Heidelberg).
- [228] Polkowski, L. and Skowron, A. (1999). Towards adaptive calculus of granules, in L. A. Zadeh and J. Kacprzyk (eds.), *Computing with Words in Information/Intelligent Systems* (Physica-Verlag, Heidelberg), pp. 201–227.

- [229] Polkowski, L. and Skowron, A. (2001). Rough mereological calculi of granules: A rough set approach to computation, *Computational Intelligence: An International Journal* **17**, 3, pp. 472–492.
- [230] Qian, Y., Wang, Q., Cheng, H., Liang, J. and Dang, C. (2015). Fuzzy-rough feature selection accelerator, *Fuzzy Sets and Systems* **258**, pp. 61–78.
- [231] Qian, Y. H., Liang, J., Pedrycz, W. and Dang, C. Y. (2011). An efficient accelerator for attribute reduction from incomplete data in rough set framework, *Pattern Recognition* **44**, pp. 1658–1670.
- [232] Quafafou, M. and Boussouf, M. (2000). Generalized rough sets based feature selection, *Intelligent Data Analysis* **4**, 1, pp. 3–17.
- [233] R. S. Michalski (1983). A theory and methodology of inductive learning, *Artificial Intelligence* **20**, pp. 111–161.
- [234] Ramsay, J. O. and Silverman, B. W. (2002). *Applied Functional Data Analysis* (Springer, Berlin).
- [235] Read, S. (1994). *Thinking about Logic: An Introduction to the Philosophy of Logic* (Oxford University Press, Oxford, New York).
- [236] Rissanen, J. (1978). Modeling by shortest data description, *Automatica* **14**, pp. 465–471.
- [237] Rissanen, J. (1985). Minimum-description-length principle, in S. Kotz and N. Johnson (eds.), *Encyclopedia of Statistical Sciences* (John Wiley & Sons, New York, NY), pp. 523–527.
- [238] Riza, L. S., Janusz, A., Bergmeir, C., Cornelis, C., Herrera, F., Ślęzak, D. and Benítez, J. M. (2014). Implementing algorithms of rough set theory and fuzzy rough set theory in the R package “roughsets”, *Information Sciences* **287**, pp. 68–89.
- [239] Roy, A. and K.Pal, S. (2003). Fuzzy discretization of feature space for a rough set classifier, *Pattern Recognition Letters* **24**, 6, pp. 895–902.
- [240] Saxena, A., Gavel, L. K. and Shrivastava, M. M. (2014). Rough sets for feature selection and classification: An overview with applications, *International Journal of Recent Technology and Engineering* **3**, pp. 62–69.
- [241] Sever, H., Raghavan, V. V. and Johnsten, T. D. (1998). The status of research on rough sets for knowledge discovery in databases, in S. Sivasundaram (ed.), *Proceedings of the Second International Conference On Non-linear Problems in Aviation and Aerospace (ICNPAA'1998), April 29-May 1, 1998, Daytona Beach, FL*, Vol. 2 (Embry-Riddle Aeronautical University, Daytona Beach, FL), pp. 673–680.
- [242] Shan, N. and Ziarko, W. (1994). An incremental learning algorithm for constructing decision rules, in W. Ziarko (ed.), *Rough Sets, Fuzzy Sets and Knowledge Discovery* (Springer Verlag, Berlin), pp. 326–334.
- [243] Shankar, B. U. (2004). Novel classification and segmentation techniques with application to remotely sensed images, in [215], pp. 295–380.
- [244] Skowron, A. (1993). Boolean reasoning for decision rules generation, in J. Komorowski and Z. W. Raś (eds.), *Proceedings of ISMIS'1993, Trondheim, Norway, June 15-18, 1993, Lecture Notes in Artificial Intelligence*, Vol. 689 (Springer-Verlag), pp. 295–305.
- [245] Skowron, A. (1995). Extracting laws from decision tables, *Computational Intelligence: An International Journal* **11**, pp. 371–388.

- [246] Skowron, A. (2000). Rough sets in KDD - plenary talk, in Z. Shi, B. Faltings and M. Musen (eds.), *16-th World Computer Congress (IFIP'2000): Proceedings of Conference on Intelligent Information Processing (IIP'2000)* (Publishing House of Electronic Industry, Beijing), pp. 1–14.
- [247] Skowron, A. (2001a). Rough sets and boolean reasoning, in W. Pedrycz (ed.), *Granular Computing: an Emerging Paradigm, Studies in Fuzziness and Soft Computing*, Vol. 70 (Springer-Verlag/Physica-Verlag, Heidelberg), pp. 95–124.
- [248] Skowron, A. (2001b). Toward intelligent systems: Calculi of information granules, *Bulletin of the International Rough Set Society* **5**, 1-2, pp. 9–30.
- [249] Skowron, A. (2004). Approximate reasoning in distributed environments, in [341], pp. 433–474.
- [250] Skowron, A. (2005a). Perception logic in intelligent systems (keynote talk), in S. Blair et al (ed.), *Proceedings of the 8th Joint Conference on Information Sciences (JCIS 2005), July 21-26, 2005, Salt Lake City, Utah, USA* (X-CD Technologies: A Conference & Management Company, 15 Coldwater Road, Toronto, Ontario, M3B 1Y8), pp. 1–5.
- [251] Skowron, A. (2005b). Rough sets and vague concepts, *Fundamenta Informaticae* **64**, 1-4, pp. 417–431.
- [252] Skowron, A. (2005c). Rough sets in perception-based computing (keynote talk), in [187], pp. 21–29.
- [253] Skowron, A. and Grzymała-Busse, J. W. (1994). From rough set theory to evidence theory, in R. Yager, M. Fedrizzi and J. Kacprzyk (eds.), *Advances in the Dempster-Shafer Theory of Evidence* (John Wiley & Sons, New York, NY), pp. 193–236.
- [254] Skowron, A., Jankowski, A. and Swiniarski, R. W. (2015). Foundations of rough sets, in R. Słowiński and Y. Yao (eds.), *Handbook of Computational Intelligence, Part C* (Springer, Heidelberg), pp. 331–348.
- [255] Skowron, A. and Nguyen, H. S. (1999). Boolean reasoning scheme with some applications in data mining, in *Proceedings of the 3-rd European Conference on Principles and Practice of Knowledge Discovery in Databases, Lecture Notes in Computer Science*, Vol. 1704 (Springer Verlag, Berlin), pp. 107–115.
- [256] Skowron, A. and Pal, S. K. (eds.) (2003). *Special volume: Rough sets, pattern recognition and data mining, Pattern Recognition Letters*, Vol. 24(6).
- [257] Skowron, A., Pal, S. K. and Nguyen, H. S. (eds.) (2011). *Special issue on Rough sets and fuzzy sets in natural computing, Theoretical Computer Science*, Vol. 412(42).
- [258] Skowron, A., Pawlak, Z., Komorowski, J. and Polkowski, L. (2002). A rough set perspective on data and knowledge, in W. Kloesgen and J. Źytkow (eds.), *Handbook of KDD* (Oxford University Press, Oxford), pp. 134–149.
- [259] Skowron, A. and Peters, J. (2003). Rough sets: Trends and challenges, in *Proceedings of the 9-th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2003), Chongqing, China, May 26-29, 2003*, pp. 25–34, (plenary talk).
- [260] Skowron, A. and Rauszer, C. (1992). The discernibility matrices and functions in information systems, in [287], pp. 331–362.

- [261] Skowron, A. and Stepaniuk, J. (1996). Tolerance approximation spaces, *Fundamenta Informaticae* **27**, 2-3, pp. 245–253.
- [262] Skowron, A. and Stepaniuk, J. (2001). Information granules: Towards foundations of granular computing, *International Journal of Intelligent Systems* **16**, 1, pp. 57–86.
- [263] Skowron, A. and Stepaniuk, J. (2004). Information granules and rough-neural computing, in [194], pp. 43–84.
- [264] Skowron, A. and Stepaniuk, J. (2005). Ontological framework for approximation, in *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2005), Regina, Canada, August 31-September 3, 2005, Part I*, pp. 718–727.
- [265] Skowron, A. and Stepaniuk, J. (2010). Approximation spaces in rough-granular computing, *Fundamenta Informaticae* **100**, pp. 141–157.
- [266] Skowron, A., Stepaniuk, J., Peters, J. and Swiniarski, R. (2006). Calculi of approximation spaces, *Fundamenta Informaticae* **72**, pp. 363–378.
- [267] Skowron, A., Stepaniuk, J. and Swiniarski, R. (2012). Modeling rough granular computing based on approximation spaces, *Information Sciences* **184**, pp. 20–43.
- [268] Skowron, A. and Suraj, Z. (1993a). A rough set approach to real-time state identification, *Bulletin of the EATCS* **50**, pp. 264–275.
- [269] Skowron, A. and Suraj, Z. (1993b). Rough sets and concurrency, *Bulletin of the Polish Academy of Sciences, Technical Sciences* **41**, pp. 237–254.
- [270] Skowron, A. and Suraj, Z. (1995). Discovery of concurrent data models from experimental tables: A rough set approach, in *Proceedings of the First International Conference on Knowledge Discovery and Data Mining (KDD 1995), Montreal, Canada, August 20-21, 1995* (AAAI Press, Menlo Park, CA), pp. 288–293.
- [271] Skowron, A. and Suraj, Z. (eds.) (2013). *Rough Sets and Intelligent Systems. Professor Zdzisław Pawlak in Memoriam*, Series Intelligent Systems Reference Library (Springer).
- [272] Skowron, A. and Swiniarski, R. (2001). Rough sets in pattern recognition, in [184], pp. 385–425.
- [273] Skowron, A. and Swiniarski, R. (2005). Rough sets and higher order vagueness, in *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2005), Regina, Canada, August 31-September 3, 2005, Part I*, pp. 33–42.
- [274] Skowron, A., Swiniarski, R. and Synak, P. (2005). Approximation spaces and information granulation, in J. F. Peters and A. Skowron (eds.), *Transactions on Rough Sets III: Journal Subline*, LNCS 3400, pp. 175–189.
- [275] Skowron, A. and Szczuka, M. (2009). Toward interactive computations: A rough-granular approach, in J. Koronacki, Z. Raś, S. Wierzchoń and J. Kacprzyk (eds.), *Advances in Machine Learning II: Dedicated to the Memory of Professor Ryszard S. Michalski, Studies in Computational Intelligence*, Vol. 263 (Springer, Heidelberg), pp. 23–42.
- [276] Skowron, A. and Wasilewski, P. (2011a). Information systems in modeling interactive computations on granules, *Theoretical Computer Science* **412**, 42, pp. 5939–5959.

- [277] Skowron, A. and Wasilewski, P. (2011b). Toward interactive rough-granular computing, *Control & Cybernetics* **40**, 2, pp. 1–23.
- [278] Ślęzak, D. (1996). Approximate reducts in decision tables, in *Sixth International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems IPMU'1996*, Vol. III (Granada, Spain), pp. 1159–1164.
- [279] Ślęzak, D. (2005). Association reducts: A framework for mining multi-attribute dependencies, in M.-S. Hacid, N. V. Murray, Z. W. Ras and S. Tsumoto (eds.), *Foundations of Intelligent Systems, 15th International Symposium on Methodologies for Intelligent Systems (ISMIS 2005), Saratoga Springs, NY, May 25-28, 2005, Lecture Notes in Artificial Intelligence*, Vol. 3488 (Springer, Heidelberg), pp. 354–363.
- [280] Ślęzak, D. (2009). Rough sets and functional dependencies in data: Foundations of association reducts, *Transactions on Computational Science* **5**, pp. 182–205.
- [281] Ślęzak, D. and Eastwood, V. (2009). Data warehouse technology by info-bright, in U. Çetintemel, S. B. Zdonik, D. Kossmann and N. Tatbul (eds.), *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD 2009)*, Providence, Rhode Island, USA, June 29 - July 2, 2009 (ACM), pp. 841–846.
- [282] Ślęzak, D., Wróblewski, J., Eastwood, V. and Synak, P. (2008). Bright-house: an analytic data warehouse for ad-hoc queries, *PVLDB* **1**, 2, pp. 1337–1345.
- [283] Ślęzak, D. (2000a). Normalized decision functions and measures for inconsistent decision tables analysis, *Fundamenta Informaticae* **44**, pp. 291–319.
- [284] Ślęzak, D. (2000b). Various approaches to reasoning with frequency-based decision reducts: A survey, in [223], pp. 235–285.
- [285] Ślęzak, D. (2002). Approximate entropy reducts, *Fundamenta Informaticae* **53**, pp. 365–387.
- [286] Ślęzak, D. and Wasilewski, P. (2007). Foundations of rough sets from vagueness perspective, in A. E. Hassanien, Z. Suraj, D. Ślęzak and P. Lingras (eds.), *Rough Computing: Theories, Technologies and Applications* (IGI Global, Hershey, PA), pp. 1–37.
- [287] Słowiński, R. (ed.) (1992). *Intelligent Decision Support - Handbook of Applications and Advances of the Rough Sets Theory, System Theory, Knowledge Engineering and Problem Solving*, Vol. 11 (Kluwer Academic Publishers, Dordrecht, The Netherlands).
- [288] Słowiński, R., Greco, S. and Matarazzo, B. (2015). Rough set methodology for decision aiding, in R. Słowiński and Y. Yao (eds.), *Handbook of Computational Intelligence, Part C* (Springer, Heidelberg), pp. 349–370.
- [289] Staab, S. and Studer, R. (eds.) (2004). *Handbook on Ontologies*, International Handbooks on Information Systems (Springer, Heidelberg).
- [290] Stawicki, S. and Ślęzak, D. (2013). Recent advances in decision bireducts: Complexity, heuristics and streams, in P. Lingras, M. Wolski, C. Cornelis, S. Mitra and P. Wasilewski (eds.), *Proceedings of the 8th International Conference on Rough Sets and Knowledge Technology (RSKT 2013)*, Halifax,

- NS, Canada, October 11-14, 2013, Lecture Notes in Computer Science, Vol. 8171 (Springer, Heidelberg), pp. 200–212.
- [291] Stepaniuk, J. (1998). Approximation spaces, reducts and representatives, in [227], pp. 109–126.
 - [292] Stepaniuk, J. (2000). Knowledge discovery by application of rough set models, in [223], pp. 137–233.
 - [293] Stepaniuk, J. (ed.) (2008a). *Rough-Granular Computing in Knowledge Discovery and Data Mining*, Studies in Computational Intelligence (Springer, Heidelberg).
 - [294] Stone, P. (2000). *Layered Learning in Multi-Agent Systems: A Winning Approach to Robotic Soccer* (The MIT Press, Cambridge, MA).
 - [295] Sun, L., Xu, J. and Tian, Y. (2012). Feature selection using rough entropy-based uncertainty measures in incomplete decision systems, *Journal of Computational Information Systems* **36**, pp. 206–216.
 - [296] Suraj, Z. (1996). Discovery of concurrent data models from experimental tables: A rough set approach, *Fundamenta Informaticae* **28**, 3-4, pp. 353–376.
 - [297] Suraj, Z. (2000). Rough set methods for the synthesis and analysis of concurrent processes, in [223], pp. 379–488.
 - [298] Suraj, Z. (2009). Discovering concurrent process models in data: A rough set approach, in *Proceedings of the 12th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing (RSFDGrC'2009)*, Delhi, India, December 15-18, 2009, pp. 12–19.
 - [299] Suraj, Z. and Pancerz, K. (2003). A synthesis of concurrent systems: A rough set approach, in *Proceedings of the 9-th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2003)*, Chongqing, China, May 26–29, 2003, pp. 299–302.
 - [300] Suraj, Z. and Pancerz, K. (2005). The rosecon system - a computer tool for modelling and analysing of processes, in *2005 International Conference on Computational Intelligence for Modelling Control and Automation (CIMCA 2005)*, International Conference on Intelligent Agents, Web Technologies and Internet Commerce (IAWTIC 2005), 28-30 November 2005, Vienna, Austria (IEEE Computer Society), pp. 829–834.
 - [301] Suraj, Z., Pancerz, K. and Owsiany, G. (2005). On consistent and partially consistent extensions of information systems, in *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2005)*, Regina, Canada, August 31-September 3, 2005, Part I, pp. 224–233.
 - [302] Swiniarski, R. (1999). Rough sets and principal component analysis and their applications. data model building and classification, in [196], pp. 275–300.
 - [303] Swiniarski, R. (2001). An application of rough sets and Haar wavelets to face recognition, in [345], pp. 561–568.
 - [304] Swiniarski, R. and Hargis, L. (1998). A new halftoning method based on error diffusion with rough set filterin, in [227], pp. 336–342.
 - [305] Swiniarski, R. and Skowron, A. (2003). Rough set methods in feature selection and extraction, *Pattern Recognition Letters* **24**, 6, pp. 833–849.

- [306] Swiniarski, R. W., Pancerz, K. and Suraj, Z. (2005). Prediction of model changes of concurrent systems described by temporal information systems, in *Proceedings of The 2005 International Conference on Data Mining (DMIN 2005), Las Vegas, Nevada, USA, June 20-23, 2005* (CSREA Press), pp. 51–57.
- [307] Swiniarski, R. W. and Skowron, A. (2004). Independent component analysis, principal component analysis and rough sets in face recognition. in [215], pp. 392–404.
- [308] Synak, P., Bazan, J. G., Skowron, A. and Peters, J. F. (2005). Spatio-temporal approximate reasoning over complex objects, *Fundamenta Informaticae* **67**, 1-3, pp. 249–269.
- [309] Szczyka, M., Skowron, A. and Stepaniuk, J. (2011). Function approximation and quality measures in rough-granular systems, *Fundamenta Informaticae* **109**, 3-4, pp. 339–354.
- [310] Thiele, L. P. (2010). *The Heart of Judgment: Practical Wisdom, Neuroscience, and Narrative* (Cambridge University Press, Cambridge, UK).
- [311] Torra, V. and Narukawa, Y. (2007). *Modeling Decisions Information Fusion and Aggregation Operators* (Springer).
- [312] Tsumoto, S. (1998). Empirical induction on medical system expert rules based on rough set theory, in [225], pp. 307–323.
- [313] Tsumoto, S., Słowiński, R., Komorowski, J. and Grzymała-Busse, J. (eds.) (2004). *Proceedings of the 4th International Conference on Rough Sets and Current Trends in Computing (RSCTC'2004), Uppsala, Sweden, June 1-5, 2004, Lecture Notes in Artificial Intelligence*, Vol. 3066 (Springer-Verlag, Heidelberg).
- [314] Tsumoto, S. and Tanaka, H. (1995). PRIMEROSE: Probabilistic rule induction method based on rough sets and resampling methods, *Computational Intelligence: An International Journal* **11**, pp. 389–405.
- [315] Unnikrishnan, K. P., Ramakrishnan, N., Sastry, P. S. and Uthurusamy, R. (2006). Network reconstruction from dynamic data, *SIGKDD Explorations* **8**, 2, pp. 90–91.
- [316] van der Aalst, W. M. P. (ed.) (2011). *Process Mining Discovery, Conformance and Enhancement of Business Processes* (Springer, Heidelberg).
- [317] Vapnik, V. (1998). *Statistical Learning Theory* (John Wiley & Sons, New York, NY).
- [318] Wang, C., He, Q., Chen, D. and Hu, Q. (2014). A novel method for attribute reduction of covering decision systems, *Information Sciences* **254**, pp. 181–196.
- [319] Wang, G. Y., Zhao, J. and An, J. J. (2005). A comparative study of algebra viewpoint and information viewpoint in attribute reduction, *Fundamenta Informaticae* **68**, pp. 289–301.
- [320] Wang, J., Jia, C. and Zhao, K. (2001). Investigation on AQ11, ID3 and the principle of discernibility matrix, *Journal of Computer Science and Technology* **16**, 1, pp. 1–12.
- [321] Wojna, A. (2005). Analogy based reasoning in classifier construction, *Transactions on Rough Sets IV: Journal Subline*, LNCS 3700, pp. 277–374.

- [322] Wong, S. K. M. and Ziarko, W. (1987). Comparison of the probabilistic approximate classification and the fuzzy model, *Fuzzy Sets and Systems* **21**, pp. 357–362.
- [323] Wróblewski, J. (1996). Theoretical foundations of order-based genetic algorithms, *Fundamenta Informaticae* **28**, pp. 423–430.
- [324] Wróblewski, J. (2000). Analyzing relational databases using rough set based methods, in *Eighth International Conference on Processing and Management of Uncertainty in Knowledge-Based Systems IPMU*, Vol. I (Madrid, Spain), pp. 256–262.
- [325] Wróblewski, J. (2004). Adaptive aspects of combining approximation spaces, in [194], pp. 139–156.
- [326] Wu, F. X. (2007). Inference of gene regulatory networks and its validation, *Current Bioinformatics* **2**, 2, pp. 139–144.
- [327] Xi, D., Wang, G., Zhang, X. and Zhang, F. (2014). Parallel attribute reduction based on MapReduce, in D. Miao, W. Pedrycz, D. Ślęzak, G. Peters, Q. Hu and R. Wang (eds.), *Proceedings of the 9th International Conference on Rough Sets and Knowledge Technology RSkt 2014, Shanghai, China, October 24–26, 2014, Lecture Notes in Computer Science*, Vol. 8818 (Springer), pp. 631–641.
- [328] Yao, J. T., Ciucci, D. and Zhang, Y. (2015a). Generalized rough sets, in R. Słowiński and Y. Yao (eds.), *Handbook of Computational Intelligence, Part C* (Springer, Heidelberg), pp. 413–424.
- [329] Yao, Y., Słowiński, R. and Greco, S. (2015b). Probabilistic rough sets, in R. Słowiński and Y. Yao (eds.), *Handbook of Computational Intelligence, Part C* (Springer, Heidelberg), pp. 387–411.
- [330] Yao, Y. Y. (1998). Generalized rough set models, in [226], pp. 286–318.
- [331] Yao, Y. Y. (2001). Information granulation and rough set approximation, *International Journal of Intelligent Systems* **16**, pp. 87–104.
- [332] Yao, Y. Y. (2003a). On generalizing rough set theory, in *Proceedings of the 9-th International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (RSFDGrC'2003), Chongqing, China, May 26–29, 2003*, pp. 44–51.
- [333] Yao, Y. Y. (2003b). Probabilistic approaches to rough sets, *Expert Systems* **20**, pp. 287–297.
- [334] Yao, Y. Y., Wong, S. K. M. and Lin, T. Y. (1997). A review of rough set models, in [121], pp. 47–75.
- [335] Yu, H., Wang, G. and Lan, F. (2011). Solving the attribute reduction problem with ant colony optimization, *Transactions on Rough Sets XIII: Journal Subline*, LNCS 5306, pp. 240–259.
- [336] Zadeh, L. A. (1965). Fuzzy sets, *Information and Control* **8**, pp. 338–353.
- [337] Zadeh, L. A. (1994). Fuzzy logic, neural networks, and soft computing, *Communications of the ACM* **37**, pp. 77–84.
- [338] Zadeh, L. A. (2001). A new direction in AI: Toward a computational theory of perceptions, *AI Magazine* **22**, 1, pp. 73–84.
- [339] Zhang, Q., Wang, J., Wang, G. and Yu, H. (2015). The approximation set of a vague set in rough approximation space, *Information Sciences* **300**, pp. 1–19.

- [340] Zheng, Z. and Wang, G. (2004). RRIA: A rough set and rule tree based incremental knowledge acquisition algorithm, *Fundamenta Informaticae* **59**, 2-3, pp. 299–313.
- [341] Zhong, N. and Liu, J. (eds.) (2004). *Intelligent Technologies for Information Analysis* (Springer, Heidelberg).
- [342] Zhu, W. (2007). Topological approaches to covering rough sets, *Information Sciences* **177**, pp. 1499–1508.
- [343] Ziarko, W. (1993). Variable precision rough set model, *Journal of Computer and System Sciences* **46**, pp. 39–59.
- [344] Ziarko, W. (2001). Probabilistic decision tables in the variable precision rough set model, *Computational Intelligence* **17**, pp. 593–603.
- [345] Ziarko, W. and Yao, Y. (eds.) (2001). *Proceedings of the 2nd International Conference on Rough Sets and Current Trends in Computing (RSCTC'2000), Banff, Canada, October 16-19, 2000, Lecture Notes in Artificial Intelligence*, Vol. 2005 (Springer-Verlag, Heidelberg).

Chapter 12

The Twin SVM Minimizes the Total Risk

Jayadeva¹, Sumit Soman¹ and Suresh Chandra²

¹*Department of Electrical Engineering
Indian Institute of Technology, Delhi, India
jayadeva@ee.iitd.ac.in, sumit.soman@gmail.com*

²*Department of Mathematics
Indian Institute of Technology, Delhi, India
chandras@maths.iitd.ac.in*

The Twin SVM (TWSVM) finds two non-parallel hyperplanes that pass through samples of the two respective classes of a binary classification problem, while lying at a distance of at least 1 from the other class. This involves solving two smaller sized Quadratic Programming Problems (QPPs) in comparison to the original SVM, and offers the advantage of insensitivity to imbalance in class sizes. This chapter points out another advantage of the TWSVM. We show that the TWSVM finds hyperplanes with a small VC dimension; in effect, the objective function of the TWSVM minimizes a linear combination of the structural and empirical risk, i.e., the TWSVM minimizes the total risk.

12.1. Introduction

The Twin Support Vector Machine (TWSVM) [1] has been a popular machine learning algorithm that has found diverse applications. The key difference between the conventional SVM and the TWSVM is that the latter solves two smaller sized QPPs to determine two non-parallel hyperplanes, each of which is as close to points of a class, and as far apart from points of the other class. The TWSVM predicts the class of test samples by computing its distance to the two hyperplanes determined from the solution of the TWSVM QPPs, and assigns the label of the class to the hyperplane to which the test point's distance is the minimum. In effect, this provides an efficient

prediction mechanism as a result of the use of non-parallel hyperplanes, as opposed to the standard SVM which determines a single maximum-margin separating hyperplane and assigns labels to test points based on the sign of the distance computed from the separating hyperplane.

Though SVMs have been fairly popular, the drawbacks of SVMs include their sensitivity to outliers and the inability to generalize well for imbalanced classification problems. Conventionally, approaches such as one-v/s-one or one-v/s-rest have been developed to use SVMs in multi-class classification scenarios, which extends the idea to use multiple binary classifiers for multi-class classification. Particularly, the one-v/s-rest approach results in a highly imbalanced binary classification problems, and the SVM model generated on such a dataset tends to be biased towards the class with larger number of samples. This effect tends to dominate with increasing number of classes, which leads to poor classification accuracy. However, the TWSVM is particularly suited to such a scenario, as each of the smaller QPPs solved by it try to maximize the distance from points of the other class, while trying to minimize the distance with points of the same class. Thus, binary classifiers built using the TWSVM on unbalanced datasets are superior to standard SVM models. Another advantage obtained by using the TWSVM is the speed of solving the optimization problem, when compared to the SVM. This is because the individual QPPs use only the training samples of a single class as opposed to the entire training dataset, and hence the separating hyperplanes are obtained faster when compared to the SVM which solves a QPP on the entire dataset. This is of significance for large datasets. These advantages have made the TWSVM very attractive in many domains. In May 2016, the TWSVM had approximately 470 citations on Google.

12.2. The Twin SVM Formulation

We now formally introduce the TWSVM formulation. Let the training set T_C for a binary data classification problem be denoted by

$$T_C = \{(x^{(i)}, y_i), x^{(i)} \in \mathbb{R}^n, y_i \in \{-1, +1\}, (i = 1, 2, \dots, m)\}. \quad (12.1)$$

Let there be m_1 patterns in class (+1) and m_2 patterns in class (-1) with $m_1 + m_2 = m$. We form the $(m_1 \times n)$ matrix A with its i^{th} row as $(x^{(i)})^T$ for which $y_i = +1$. The $(m_2 \times n)$ matrix B is constructed similarly with data points $x^{(i)}$ for which $y_i = -1$.

The linear TWSVM formulation consists of determining two hyperplanes $x^T w^{(1)} + b^{(1)} = 0$ and $x^T w^{(2)} + b^{(2)} = 0$ by solving the following pair

of QPPs:

$$(TWSVM1) \quad \underset{(w^{(1)}, b^{(1)}, q^{(1)})}{\text{Min}} \quad \frac{1}{2} \| (Aw^{(1)} + e_1 b^{(1)}) \|_2 + C_1 e_2^T q^{(1)}$$

s.t.

$$\begin{aligned} -(Bw^{(1)} + e_2 b^{(1)}) + q^{(1)} &\geq e_2, \\ q^{(1)} &\geq 0, \end{aligned} \quad (12.2)$$

and,

$$(TWSVM2) \quad \underset{(w^{(2)}, b^{(2)}, q^{(2)})}{\text{Min}} \quad \frac{1}{2} \| (Bw^{(2)} + e_2 b^{(2)}) \|_2 + C_2 e_1^T q^{(2)}$$

s.t.

$$\begin{aligned} (Aw^{(2)} + e_1 b^{(2)}) + q^{(2)} &\geq e_1, \\ q^{(2)} &\geq 0. \end{aligned} \quad (12.3)$$

Here $w^{(1)} \in \mathbb{R}^n, w^{(2)} \in \mathbb{R}^n, b^{(1)} \in \mathbb{R}, b^{(2)} \in \mathbb{R}, q^{(1)} \in \mathbb{R}^{m_2}$ and $q^{(2)} \in \mathbb{R}^{m_1}$. Also $C_1 > 0, C_2 > 0$ are parameters, and $e_1 \in \mathbb{R}^{m_1}, e_2 \in \mathbb{R}^{m_2}$ are vectors of ‘ones’, i.e. each component is ‘1’ only.

Let us examine the formulation of (TWSVM1) in (12.2). The first term in the objective function of (TWSVM1) is the sum of squared distances from the hyperplane $x^T w^{(1)} + b^{(1)} = 0$ to points of class (+1). Therefore, its minimization tends to keep the hyperplane $x^T w^{(1)} + b^{(1)} = 0$ close to the points of class (+1). The constraints in (12.2) require the hyperplane $x^T w^{(1)} + b^{(1)} = 0$ to be at a distance of at least 1 from points of class (-1). The slack variables q_i are used to measure the error wherever the hyperplane is closer to samples of class (-1), than the specified minimum distance of 1. The second term of the objective function of (TWSVM1) tries to minimize mis-classification due to points belonging to class (-1). The positive hyper-parameter C_1 determines the relative emphasis of the mis-classification component of the objective function. The formulation (TWSVM2) in (12.3) may be explained in a similar way.

Several extensions of the TWSVM have also been proposed such as Twin Bounded SVMs [2], Coordinate Descent Margin-based Twin SVM [3], Twin SVM with probabilistic outputs [4], Least-squares Twin SVM [5], Least-squares Recursive Projection Twin SVMs [6], Knowledge based Least Squares Twin SVM [7], Margin Based Twin SVM [8], ϵ -Twin SVM [9], Twin Parametric Margin Classifier [10], Least-squares Twin Parametric Margin Classifier [11], Twin Support Vector Hypersphere Classifier [12], Least-squares Twin Support Vector Hypersphere Classifier [13], Structural Twin SVM [14] among others. The Twin SVM has also been used for

regression [15, 16], clustering [17] and has proven to be efficient even in the primal formulation [18, 19]. A comprehensive review of the variants and applications of Twin SVM has been presented by Ding *et al.* [20, 21], Tian *et al.* [22] and Tomar *et al.* [23].

It has also been used successfully for several machine learning problems, including classification of bio-medical data such as Electroencephalogram (EEG) [24] and surface Electromyogram (sEMG) [25–27]. Systems have also been developed for clustered microcalcification detection (MCs) [28–30], action recognition [31, 32], intrusion detection [33–35], license plate recognition [36], Parkinson’s [37] and heart [38] disease diagnosis system, star spectra data classification [39], defect diagnosis in steel surface [40, 41], rolling bearing [42], computer software [43], wireless sensor nets [44], image de-noising [45], economy development prediction [46], particle swarm optimization [47, 48], speaker recognition [49, 50] among others.

12.3. The Twin SVM tries to Find a Gap Tolerant Classifier with Minimum VC Dimension

We first observe that the TWSVM is a gap tolerant classifier, and use results on fat margin classifier to show that the TWSVM formulation tries to minimize the sum of a structural risk, and an empirical risk component. A fat margin classifier, also termed as a gap tolerant classifier, is a hyperplane classifier with a specified margin Δ with the following provision: we ignore errors made by the classifier in the margin area, but count errors that are made outside it. Figure 12.1 illustrates this notion.

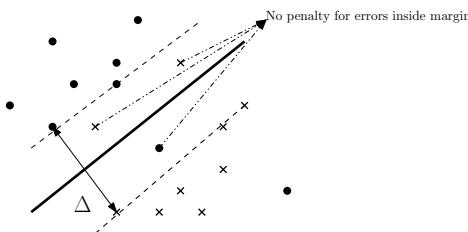


Fig. 12.1. A gap-tolerant classifier.

We now observe that the first hyperplane $x^T w^{(1)} + b^{(1)} = 0$ is the decision boundary of a fat margin classifier that ignores any errors made with regard to training samples of class (+1); this is illustrated in Fig. 12.2. The optimization problem (12.2) that is used to determine this classifier

only counts errors made with regard to samples of class (+1). Similarly, the second hyperplane $x^T w^{(2)} + b^{(2)} = 0$ is the decision boundary of a fat margin classifier that ignores any errors made with regard to training samples of class (-1). Of late, there has been much interest in learning machines with

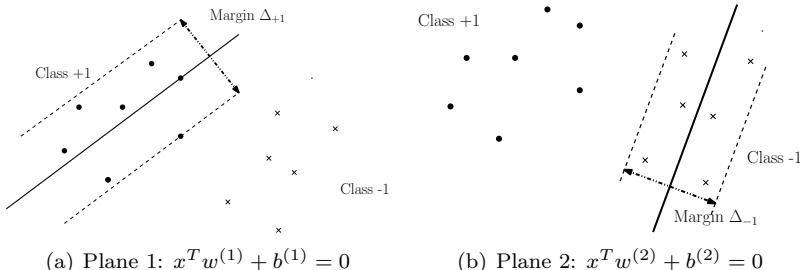


Fig. 12.2. The Twin SVM as a gap-tolerant classifier.

minimum VC dimension. A small VC dimension leads to lower error rates on test data, and leads to good generalization. The Minimal Complexity Machine (MCM) [51] shows that it is possible to learn a hyperplane classifier with a small VC dimension by solving a Linear Programming Problem (LPP). Taking hint from the MCM [51], and from Burges's tutorial [52], we note that the VC dimension γ of a gap tolerant hyperplane classifier with margin $d \geq d_{min}$ is bounded by [53]

$$\gamma \leq 1 + \text{Min}\left(\frac{R^2}{d_{min}^2}, n\right) \quad (12.4)$$

where R denotes the radius of the smallest sphere enclosing all the training samples. Burges, in [52], stated that “*the above arguments strongly suggest that algorithms that minimize $\frac{R^2}{d^2}$ can be expected to give better generalization performance.*” Further evidence for this is found in the following theorem of (Vapnik, 1998), which we quote without proof.

Consider a hyperplane $u^T x + v = 0$. From [51], we know that the VC dimension of this hyperplane classifier is upper bounded by h^2 , i.e.

$$\gamma \leq h^2, \text{ where } h = \frac{\text{Max}_{i=1,2,\dots,M} \|u\| \|x^i\|}{\text{Min}_{i=1,2,\dots,M} \|u^T x^i + v\|} \quad (12.5)$$

The VC dimension bound is best interpreted for the linearly separable case, where all samples are at a distance of at least 1 from the separating

hyperplane, i.e.,

$$u^T x + v \geq 1 \text{ for all samples of class (+1)} \quad (12.6)$$

and

$$u^T x + v \leq -1 \text{ for all samples of class (-1)} \quad (12.7)$$

$$\text{i.e. } \|u^T x^i + v\| \geq 1, \quad i = 1, 2, \dots, m \quad (12.8)$$

$$\text{or } \underset{i=1,2,\dots,M}{\text{Min}} \|u^T x^i + v\| = 1 \quad (12.9)$$

Hence,

$$h^2 = \underset{i=1,2,\dots,M}{\text{Max}} \|u\|^2 \|x^i\|^2 \leq \sum_i (u^T x^i)^2 \quad (12.10)$$

In the TWSVM, consider the first hyperplane $x^T w^{(1)} + b^{(1)} = 0$. This hyperplane is determined by solving (12.2), where the objective function is given by

$$\frac{1}{2} \|(Aw^{(1)} + e_1 b^{(1)})\|^2 \quad (12.11)$$

and is identical to (12.10) with the summation taken over samples of class (+1). Similarly, the second hyperplane $x^T w^{(2)} + b^{(2)} = 0$ is determined by solving (12.3), where the objective function is given by

$$\frac{1}{2} \|(Bw^{(2)} + e_2 b^{(2)})\|^2 \quad (12.12)$$

and is identical to (12.10) with the summation taken over samples of class (-1).

In other words, the first term of the objective function of each of the problems in the TWSVM, namely (12.2) and (12.3), attempts to minimize an upper bound on the VC dimension of the corresponding hyperplane classifier. The second term of the objective function is a measure of the mis-classification error over samples that do not lie in the margin, and is summed over all the training samples of the other class. Hence, the TWSVM objective function tries to minimize a sum of a term that is related to the structural risk, and another that depends on the empirical error. Hence, the TWSVM can be thought of as an approach that attempts to reduce the total risk. Note that this is an approximation to the total risk. Vapnik and Chervonenkis [54] deduced, that with probability $(1 - \eta)$,

$$R_{total}(\lambda) \leq R_{emp}(\lambda) + \sqrt{\frac{h(\ln \frac{2m}{h} + 1) - \ln \frac{\eta}{4}}{m}} \quad (12.13)$$

$$\text{where, } R_{emp}(\lambda) = \frac{1}{m} \sum_{i=1}^m |f_\lambda(x_i) - y_i|, \quad (12.14)$$

and f_λ is a function having VC-dimension h with the smallest empirical risk on a dataset $\{x_i, i = 1, 2, \dots, m\}$ of m data points with corresponding labels $\{y_i, i = 1, 2, \dots, m\}$.

Minimizing the total risk is a difficult proposition, even if one could devise a way to do that. It has been recently shown that the Minimal Complexity Machine (MCM) [51] learns a hyperplane classifier by minimizing an exact bound on the VC dimension. This chapter establishes that the TWSVM minimizes a term that is related to an upper bound on the VC dimension, while also trading off the empirical risk. In effect, the TWSVM tries to minimize the total risk. It also implies that modifications to the objective function, that have been suggested in the literature as a means to achieve structural risk minimization, are not necessary and may be done away with.

12.4. An Experimental Observation

In the derivation of (12.5), a basic assumption made is that

$$R = \underset{i=1,2,\dots,M}{\text{Max}} \|x^i\| \quad (12.15)$$

which implicitly assumes that the data is centered at the origin. It is easy to see that if this is not the case, the bound on the VC dimension would be much larger, and poorer test set accuracies might follow. We show, by means of a simple experiment, that this is indeed the case.

We ran the TWSVM on raw data. In a second set of experiments, the data samples are first zero centered, i.e. the mean of the training samples is shifted to the origin by subtracting the mean from all training samples. If our claim is justified, the TWSVM ought to show improved generalization in the second case.

Table 12.1 shows test set accuracies for a few benchmark datasets taken from the UCI machine learning repository. Accuracies were computed by using a five fold cross-validation methodology. The table shows accuracies for two cases: when the data is shifted so that the mean of shifted samples is the origin, and when the raw data is used. The results are on expected lines, the accuracies are significantly higher when the data is zero-centered. This is consistent with our understanding of the objective function of the Twin SVM: any increase in the radius of the dataset would cause the classifier to have a larger VC dimension bound, which can lead to poorer generalization.

Table 12.1. Mean Centering Results.

S. No.	Dataset	Original Accuracy	Mean Centered
1	Haberman	66.64 ± 16.46	77.77 ± 1.84
2	Heart Statlog	63.33 ± 16.46	85.55 ± 3.31
3	Transfusion	76.19 ± 4.60	80.47 ± 4.23
4	Promoters	65.15 ± 8.25	82.12 ± 3.60
5	Voting	93.56 ± 3.40	95.86 ± 1.31
6	Hepatitis	61.45 ± 7.98	70.39 ± 7.08
7	Australian	77.97 ± 11.37	86.23 ± 2.29
8	Credit Screening	69.27 ± 3.46	68.11 ± 1.69
9	German credit	70.00 ± 1.36	74.60 ± 2.22
10	Fertility Diagnosis	81.67 ± 7.63	91.67 ± 7.64
11	Hepatitis	61.45 ± 7.98	70.39 ± 7.083

12.5. Conclusion

In this chapter, we point out that the TWSVM is essentially a fat margin, or gap tolerant classifier. Using prior work in computational learning theory on gap tolerant classifiers, we show that the objective function of the TWSVM contains a term that is an upper bound on the VC dimension of a gap tolerant hyperplane classifier. This implies that the TWSVM formulation tries to minimize the sum of two terms: the first being one that is related to the structural risk, and a second term that is a measure of the empirical error. In a nutshell, the TWSVM tries to minimize the total risk.

References

- [1] Jayadeva, R. Khemchandani, and S. Chandra, Twin support vector machines for pattern classification, *Pattern Analysis and Machine Intelligence, IEEE Transactions on.* **29**(5), 905–910 (2007).
- [2] Y.-H. Shao, C.-H. Zhang, X.-B. Wang, and N.-Y. Deng, Improvements on twin support vector machines, *Neural Networks, IEEE Transactions on.* **22**(6), 962–968 (June, 2011). ISSN 1045-9227. doi: 10.1109/TNN.2011.2130540.
- [3] Y.-H. Shao and N.-Y. Deng, A coordinate descent margin based-twin support vector machine for classification, *Neural networks.* **25**, 114–121 (2012).
- [4] Y.-H. Shao, N.-Y. Deng, Z.-M. Yang, W.-J. Chen, and Z. Wang, Probabilistic outputs for twin support vector machines, *Knowledge-Based Systems.* **33**, 145–151 (2012).
- [5] M. Arun Kumar and M. Gopal, Least squares twin support vector machines for pattern classification, *Expert Systems with Applications.* **36**(4), 7535–7543 (2009).

- [6] Y.-H. Shao, N.-Y. Deng, and Z.-M. Yang, Least squares recursive projection twin support vector machine for classification, *Pattern Recognition*. **45**(6), 2299–2307 (2012).
- [7] M. Arun Kumar, R. Khemchandani, M. Gopal, and S. Chandra, Knowledge based least squares twin support vector machines, *Information Sciences*. **180**(23), 4606–4618 (2010).
- [8] Y.-H. Shao and N.-Y. Deng, A novel margin-based twin support vector machine with unity norm hyperplanes, *Neural Computing and Applications*. pp. 1–9 (2013).
- [9] Y.-H. Shao, C.-H. Zhang, Z.-M. Yang, L. Jing, and N.-Y. Deng, An ε -twin support vector machine for regression, *Neural Computing and Applications*. pp. 1–11 (2012).
- [10] X. Peng, TPM SVM: a novel twin parametric-margin support vector machine for pattern recognition, *Pattern Recognition*. **44**(10), 2678–2692 (2011).
- [11] Y.-H. Shao, Z. Wang, W.-J. Chen, and N.-Y. Deng, Least squares twin parametric-margin support vector machine for classification, *Applied Intelligence*. pp. 1–14 (2013).
- [12] X. Peng and D. Xu, Twin support vector hypersphere (TSVH) classifier for pattern recognition, *Neural Computing and Applications*. **24**(5), 1207–1220 (2014).
- [13] X. Peng, Least squares twin support vector hypersphere (LS-TSVH) for pattern recognition, *Expert Systems with Applications*. **37**(12), 8371–8378 (2010).
- [14] Z. Qi, Y. Tian, and Y. Shi, Structural twin support vector machine for classification, *Knowledge-Based Systems*. **43**, 74–81 (2013).
- [15] X. Peng, Tsvr: an efficient twin support vector machine for regression, *Neural Networks*. **23**(3), 365–372 (2010).
- [16] X. Peng, Efficient twin parametric insensitive support vector regression model, *Neurocomputing*. **79**, 26–38 (2012).
- [17] Z. Wang, Y. Shao, L. Bai, and N. Deng, Twin support vector machine for clustering., *IEEE transactions on neural networks and learning systems* (2015).
- [18] X. Peng, Building sparse twin support vector machine classifiers in primal space, *Information Sciences*. **181**(18), 3967–3980 (2011).
- [19] X. Peng, Primal twin support vector regression and its sparse approximation, *Neurocomputing*. **73**(16), 2846–2858 (2010).
- [20] S. Ding, J. Yu, B. Qi, and H. Huang, An overview on twin support vector machines, *Artificial Intelligence Review*. pp. 1–8 (2013).
- [21] S. Ding, X. Hua, and J. Yu, An overview on nonparallel hyperplane support vector machine algorithms, *Neural Computing and Applications*. **25**(5), 975–982 (2014).
- [22] Y. Tian and Z. Qi, Review on: twin support vector machines, *Annals of Data Science*. **1**(2), 253–277 (2014).
- [23] D. Tomar and S. Agarwal, Twin support vector machine: a review from 2007 to 2014, *Egyptian Informatics Journal* (2015).
- [24] S. Soman *et al.*, High performance EEG signal classification using classifiability and the twin svm, *Applied Soft Computing*. **30**, 305–318 (2015).

- [25] S. Arjunan, D. Kumar, and G. Naik. A machine learning based method for classification of fractal features of forearm semg using twin support vector machines. In *Conference proceedings:... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*, vol. 2010, pp. 4821–4824 (2009).
- [26] G. R. Naik, D. K. Kumar, et al., Twin svm for gesture classification using the surface electromyogram, *Information Technology in Biomedicine, IEEE Transactions on.* **14**(2), 301–308 (2010).
- [27] D. K. Kumar, S. Poosapadi Arjunan, and V. P. Singh, Towards identification of finger flexions using single channel surface electromyography—able bodied and amputee subjects, *Journal of neuroengineering and rehabilitation.* **10** (1), 50 (2013).
- [28] X. Zhang, X. Gao, and Y. Wang, Mcs detection with combined image features and twin support vector machines, *Journal of Computers.* **4**(3), 215–221 (2009).
- [29] X. Zhang, X. Gao, and Y. Wang, Twin support tensor machines for mcs detection, *Journal of Electronics (China).* **26**(3), 318–325 (2009).
- [30] X. Zhang. Boosting twin support vector machine approach for mcs detection. In *Information Processing, 2009. APCIP 2009. Asia-Pacific Conference on*, vol. 1, pp. 149–152 (2009).
- [31] J. A. Nasiri, N. M. Charkari, and K. Mozafari, Energy-based model of least squares twin support vector machines for human action recognition, *Signal Processing.* **104**, 248–257 (2014).
- [32] K. Mozafari, J. A. Nasiri, N. M. Charkari, and S. Jalili. Action recognition by local space-time features and least square twin svm (ls-tsvm). In *Informatics and Computational Intelligence (ICI), 2011 First International Conference on*, pp. 287–292 (2011).
- [33] J. He and S.-h. Zheng, Intrusion detection model with twin support vector machines, *Journal of Shanghai Jiaotong University (Science).* **19**, 448–454 (2014).
- [34] X. S. L. Jing, Intrusion detection based on twin svm, *Journal of Hubei University of Education.* **2**, 019 (2009).
- [35] X. Ding, G. Zhang, Y. Ke, B. Ma, and Z. Li. High efficient intrusion detection methodology with twin support vector machines. In *Information Science and Engineering, 2008. ISISE'08. International Symposium on*, vol. 1, pp. 560–564 (2008).
- [36] S. B. Gao, J. Ding, and Y. J. Zhang, License plate location algorithm based on twin svm, *Advanced Materials Research.* **271**, 118–124 (2011).
- [37] D. Tomar, B. R. Prasad, and S. Agarwal. An efficient parkinson disease diagnosis system based on least squares twin support vector machine and particle swarm optimization. In *Industrial and Information Systems (ICIIS), 2014 9th International Conference on*, pp. 1–6 (2014).
- [38] D. Tomar and S. Agarwal, Feature selection based least square twin support vector machine for diagnosis of heart disease, *International Journal of Bio-Science and Bio-Technology.* **6**(2), 69–82 (2014).
- [39] Z.-B. LIU, Y.-Y. GAO, and J.-Z. WANG, Automatic classification method of star spectra data based on manifold fuzzy twin support vector machine.

- [40] M. Chu, R. Gong, and A. Wang, Strip steel surface defect classification method based on enhanced twin support vector machine, *ISIJ international*. **54**(1), 119–124 (2014).
- [41] M. Chu, A. Wang, R. Gong, and M. Sha, Strip steel surface defect recognition based on novel feature extraction and enhanced least squares twin support vector machine, *ISIJ International*. **54**(7), 1638–1645 (2014).
- [42] Z. Shen, N. Yao, H. Dong, and Y. Yao, Application of twin support vector machine for fault diagnosis of rolling bearing. In *Mechatronics and Automatic Control Systems*, pp. 161–167. Springer (2014).
- [43] S. Agarwal, D. Tomar, et al. Prediction of software defects using twin support vector machine. In *Information Systems and Computer Networks (ISCON), 2014 International Conference on*, pp. 128–132 (2014).
- [44] M. Ding, D. Yang, and X. Li, Fault diagnosis for wireless sensor by twin support vector machine, *Mathematical Problems in Engineering*. **2013** (2013).
- [45] H.-Y. Yang, X.-Y. Wang, P.-P. Niu, and Y.-C. Liu, Image denoising using nonsubsampled shearlet transform and twin support vector machines, *Neural Networks*. **57**, 152–165 (2014).
- [46] F. Su and H. Shang, A wavelet kernel-based primal twin support vector machine for economic development prediction, *Mathematical Problems in Engineering*. **2013** (2013).
- [47] S. Ding, J. Yu, H. Huang, and H. Zhao, Twin support vector machines based on particle swarm optimization, *Journal of Computers*. **8**(9), 2296–2303 (2013).
- [48] S. Ding, F. Wu, R. Nie, J. Yu, and H. Huang, Twin support vector machines based on quantum particle swarm optimization, *Journal of Software*. **8**(7), 1743–1750 (2013).
- [49] H. Cong, C. Yang, and X. Pu. Efficient speaker recognition based on multi-class twin support vector machines and gmms. In *2008 IEEE conference on robotics, automation and mechatronics*, pp. 348–352 (2008).
- [50] C. Yang and Z. Wu. Study to multi-twin support vector machines and its applications in speaker recognition. In *Computational Intelligence and Software Engineering, 2009. CiSE 2009. International Conference on*, pp. 1–4 (2009).
- [51] Jayadeva, Learning a hyperplane classifier by minimizing an exact bound on the VC dimension, *Neurocomputing*. **149**, Part B(0), 683 – 689 (2015). ISSN 0925-2312. doi: <http://dx.doi.org/10.1016/j.neucom.2014.07.062>. URL <http://www.sciencedirect.com/science/article/pii/S0925231214010194>.
- [52] C. J. Burges, A tutorial on support vector machines for pattern recognition, *Data mining and knowledge discovery*. **2**(2), 121–167 (1998).
- [53] V. N. Vapnik, Statistical learning theory (1998).
- [54] V. N. Vapnik and A. J. Chervonenkis, Theory of pattern recognition (1974).

Chapter 13

Dynamic Kernels based Approaches to Analysis of Varying Length Patterns in Speech and Image Processing Tasks

Veena Thenkanidiyoor¹, Dileep A. D.² and C. Chandra Sekhar³

¹*Department of Computer Science and Engineering
National Institute of Technology Goa
Farmagudi, Ponda, Goa, India*

²*School of Computing and Electrical Engineering
Indian Institute of Technology Mandi
Khamand Campus, Mandi, India*

³*Department of Computer Science and Engineering
Indian Institute of Technology Madras, Chennai, India*

Varying length patterns extracted from speech and image data correspond to sets or sequences of local feature vectors. Kernels designed for varying length patterns are called as dynamic kernels. This Chapter presents the issues in designing the dynamic kernels, different methods for designing the dynamic kernels, and the suitability of dynamic kernels based approaches to speech and image processing tasks. We explore the matching based approaches to designing dynamic kernels for speech and image processing tasks. An intermediate matching kernel (IMK) for a pair of varying length patterns is constructed by matching the pairs of local feature vectors selected using a set of virtual feature vectors. For varying length patterns corresponding to sets of local feature vectors, a Gaussian mixture model (GMM) is used as the set of virtual feature vectors. The GMM-based IMK is considered for speech processing tasks such as speech emotion recognition and speaker identification, and for image processing tasks such as image classification, image matching and image annotation in content-based image retrieval. For varying length patterns corresponding to sequences of local feature vectors, a hidden Markov model (HMM) is used for selection of local feature vectors in constructing the IMK. The HMM-based IMK is considered for speech recognition tasks such as E-set recognition and Consonant-Vowel (CV) unit recognition. We present the studies comparing the IMK based ap-

proaches and the other dynamic kernels based approaches.

13.1. Introduction

Pattern analysis corresponds to automatic discovery of patterns in data where a pattern is any structure, relation or regularity present in data. Pattern discovery from bio-sequences involves classification and matching of discrete symbol sequences. Pattern discovery from multimedia data such as audio, speech, image and video involves classification and matching of sequences or sets of continuous valued local feature vectors. Pattern classification involving classification of an example to one of the predefined classes is an important pattern analysis task. Pattern classification is widely used in various applications such as biometric based authentication systems, speech recognition, speaker recognition, speech emotion recognition, gene classification, text documents organization, management of repository of multimedia data such as image, music, audio and video. Another interesting pattern analysis task is pattern matching. Matching of patterns involves computing a measure of similarity or dissimilarity between a pair of patterns, and is useful in information retrieval tasks that involve finding the data items such as images, text documents, audio or video clips that are similar to the given query.

An important issue in classification and matching of multimedia data is that the examples are often varying length in nature. In varying length data, the number of local feature vectors is different for different examples. It is required to consider varying length sequences of continuous valued local feature vectors in speech recognition, video classification and music retrieval. In image classification, speech emotion recognition and speaker recognition varying length sets of local feature vectors are considered. Conventional methods for classification of varying length patterns comprising of continuous valued local feature vectors use hidden Markov models (HMMs) or Gaussian mixture models (GMMs). Conventional methods for matching a pair of examples involve using a suitable distance measure on the varying length patterns. Kernel methods based approaches such as support vector machines and kernel based matching have been explored for classification and matching of static patterns and varying length patterns. Kernel methods have been shown to give a good generalization performance. This Chapter presents kernel methods based approaches to classification and matching of varying length patterns.

Kernel methods for pattern analysis involve performing a nonlinear

transformation from the input feature space to a higher dimensional feature space induced by a Mercer kernel function, and then constructing an optimal linear solution in the kernel feature space. Support vector machine is a kernel method for pattern classification that constructs an optimal separating hyperplane corresponding to the maximum margin hyperplane for the data of two classes in a kernel feature space [1]. The choice of the kernel function used in the kernel methods is important for their performance. Several kernel functions have been proposed for static patterns. Kernel methods for varying length pattern analysis adopt one of the following two strategies: (1) Convert a varying length pattern into a static pattern and then use a kernel function defined for static patterns, and (2) Design and use a kernel function for varying length patterns. Kernel functions designed for varying length patterns are referred to as dynamic kernels [2]. Examples of dynamic kernels for discrete sequences are the edit distance kernel and the string kernel. Examples of dynamic kernels for continuous feature vector sequences are the Fisher kernel [3], the Gaussian mixture model (GMM) supervector kernel [4] and the intermediate matching kernel [5]. This Chapter presents the dynamic kernels based approaches to classification and matching of varying length patterns.

The rest of the chapter is organized as follows: Section 13.2 discusses about representation of varying length patterns of multimedia data. Pattern classification using support vector machines (SVM) is presented in Section 13.3. Section 13.4 describes the approaches to design of dynamic kernels for varying length patterns. A review of the dynamic kernels for varying length patterns represented as sets of local feature vectors is presented in Section 13.5. Section 13.6 presents a review of dynamic kernels for varying length patterns represented as sequences of local feature vectors. Studies on classification and matching of varying length patterns represented as sets of local feature vectors are presented in Section 13.7. This section presents studies on speech emotion recognition, speaker identification, scene image classification, scene image matching and retrieval, and scene image annotation. Section 13.8 presents recognition of consonant-vowel (CV) units of speech using dynamic kernels for sequences of local feature vectors.

13.2. Representation of varying length patterns

Varying length pattern analysis involves analysis of patterns that are either discrete symbol sequences as in bio-sequence and text data, or continu-

ous valued local feature vector sequences as in speech, audio, handwritten character, image and video data. The focus of this work is on analysis of varying length patterns of speech and image data. Short-time analysis of a speech signal involves performing spectral analysis on frames of 20 milliseconds duration each and representing every frame by a real valued local feature vector. The speech signal of an utterance with T frames is represented as a sequential pattern $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t, \dots, \mathbf{x}_T)$, where $\mathbf{x}_t \in \mathbb{R}^d$ is the local feature vector for frame t . As the duration of utterances varies, the number of frames varies. Therefore, the number of local feature vectors obtained is different for different utterances. Hence, the speech signals of utterances are represented as varying length patterns. The commonly used short-time analysis based features are the Mel-frequency cepstral coefficients [6, 7], linear predictive cepstral coefficients [7, 8], line spectral frequencies [7], perceptual linear prediction coefficients [9] and group delay processing based features [10, 11].

The main issue in a classification task involving speech signals is building a classifier for varying length patterns. Acoustic modeling of subword units of speech such as phonemes, triphones and syllables, involves developing classification models for patterns extracted from speech segments of subword units. Figure 13.1 shows the speech signal waveforms of two utterances of syllable ‘ba’. Though both the signals belong to utterances of

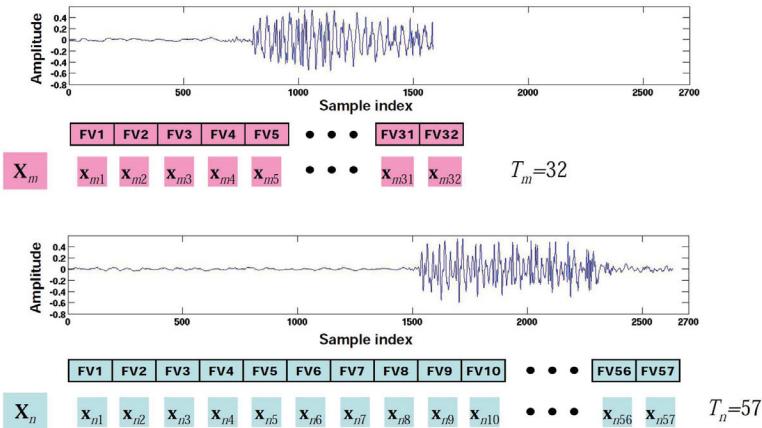


Fig. 13.1. Speech signal waveforms of two short duration utterances of syllable ‘ba’. These signals are recorded at a sampling rate of 16KHz. Each utterance is represented as a sequence of local feature vectors (FVs). The varying length patterns of utterances are denoted by X_m and X_n , with T_m and T_n denoting the number of frames respectively.

the same class, their durations are different and hence the number of local feature vectors obtained after short time analysis is different. Duration of speech signal of a subword unit is short and it is necessary to model the temporal dynamics and correlations among the features while developing the classification models for subword units. In such cases, it is necessary to represent a speech signal as a sequence of local feature vectors. Hidden Markov models (HMMs) [8] are commonly used for classification of sequences of local feature vectors. Maximum likelihood (ML) based method such as Baum-Welch method is commonly used for estimation of parameters of an HMM for each class.

In the tasks such as text-independent speaker recognition, spoken language identification and speech emotion recognition, a phrase or a sentence is used as a unit. Figure 13.2 shows the speech signal waveforms of two long duration utterances where the emotion quotient of the utterances is ‘anger’. The duration of an utterance of a phrase or a sentence is long. Preserving the sequence information in a phrase or a sentence is not considered to be critical for these tasks. The phonetic content in the speech signal is not considered to be important for these tasks. There may not

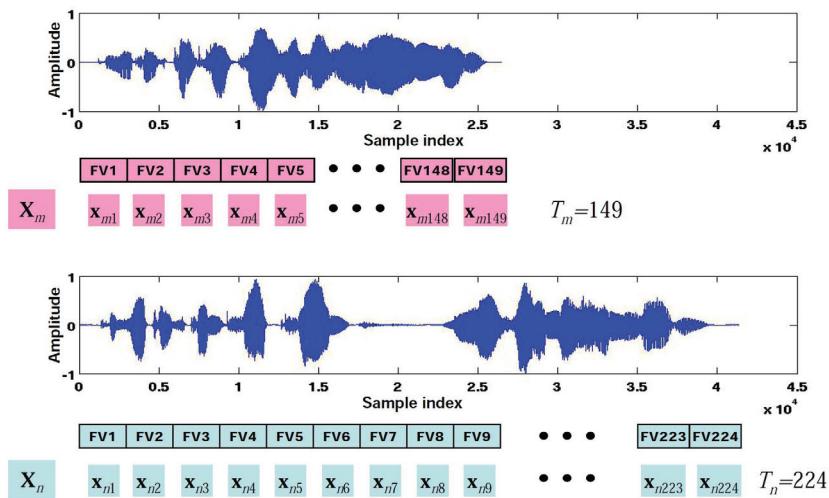


Fig. 13.2. Speech signal waveforms of two long duration utterances with ‘anger’ as emotion. These signals are recorded at a sampling rate of 16KHz. Each utterance is represented as a set of local feature vectors (FVs). The varying length patterns of utterances are denoted by X_m and X_n , with T_m and T_n denoting the number of frames respectively.

be any temporal correspondence between the frames of different utterances belonging to a class, as illustrated in Figure 13.2. In such cases, the speech signal of an utterance can be represented as a set of local feature vectors, without preserving the sequence information.

A scene image can be defined as a semantically coherent and human-scaled view of a real-world environment [12]. Scene images differ from object images in that an object image consists of a single object in a rather uniform and artificial background. Many concepts or objects co-occur in a scene image with no restriction on the scale and location of their appearance. A suitable representation for a scene image is important for any pattern analysis task involving scene images. The issues related to scene image representation correspond to the approaches to extract the visual information from an image and the approaches to represent it. Visual information of a scene image is represented using low-level features such as color and texture features that can be computed from the pixels of an image.

A low-level visual feature vector may be extracted from the entire image or from a local region of a scene image. The low-level features extracted by processing all the pixels in an image are used to obtain a \tilde{d} -dimensional global feature vector as illustrated in Figure 13.3. In [13], a scene image is represented by global texture features for classification. Here, the texture features are extracted using Gabor filters and wavelet transforms. In [14], an image is represented as a global feature vector comprising of color histogram, shape features and texture features for content based image retrieval (CBIR). In [15] and [16], a scene image is represented by a global feature vector comprising of color and texture features for CBIR. A scene image represented by a global vector corresponding to the gist of the image in [17].

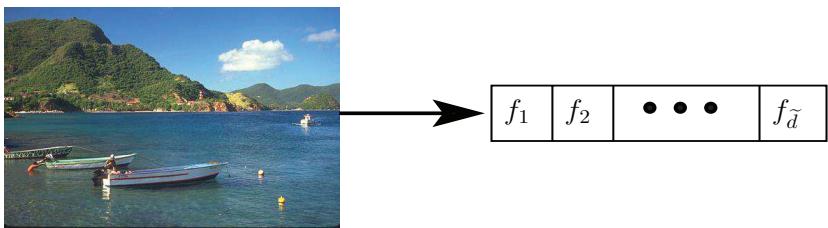


Fig. 13.3. Illustration of representing an image by a vector of global features extracted from the entire image.

Semantic variability of a scene image may not be well represented by a global feature vector. It is necessary to extract features from each of the local regions of a scene image to capture the local semantics [18–20]. An important issue in extraction of local feature vectors from a scene image is identification of suitable local regions in the image. Local regions corresponding to natural segments are expected to be associated with semantic objects or concepts in a scene image [21–25]. For this purpose, an image needs to be segmented using a suitable method. In [26], normalized cut criterion based graph partitioning approach is proposed for image segmentation. In [27], the mean shift technique, a non-parametric feature space analysis technique, is used for segmenting an image. In [28], an image is segmented using the region growing and region merging approach. From each of the segmented regions, a local feature vector is extracted as illustrated in Figure 13.4. A scene image with the manually marked region boundaries is shown in Figure 13.4 [29]. If there exists a segmentation method capable of segmenting an image into regions of concepts in a scene image perfectly as shown in Figure 13.4, then considering the natural segments is a good option for extraction of local feature vectors. However, the existing methods for image segmentation are not capable of segmenting a scene image according to its semantic content.

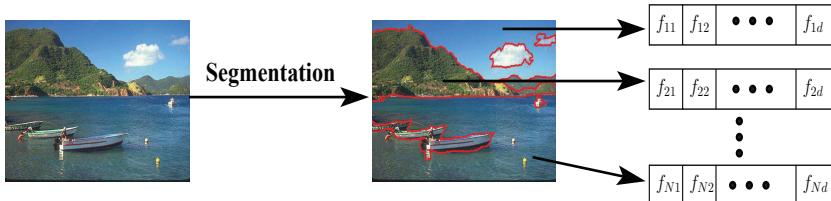


Fig. 13.4. Illustration of representing a scene image by local feature vectors that are extracted from natural segments of the image.

An alternative approach to considering a natural image segment for extraction of a local feature vector is to consider a fixed size block of a scene image to extract a local feature vector as illustrated in Figure 13.5. There are two approaches to consider fixed size blocks of a scene image for extraction of local feature vectors depending on their location. In the first approach, fixed size non-overlapping blocks are considered uniformly over the entire image as shown in Figure 13.5 [30, 31]. In the second approach, blocks are considered only in the most informative areas of an

image. The most informative areas are detected using interest points or salient points [32, 33] and a block around every interest point is considered for extraction of a local feature vector as illustrated in Figure 13.6. In this method, every scene image is represented by that many local feature vectors as that of the number of interest points in the image. In [34, 35], it is shown that representing a scene image using local feature vectors extracted from non-overlapping blocks of a scene image is good enough for pattern analysis tasks involving scene images.

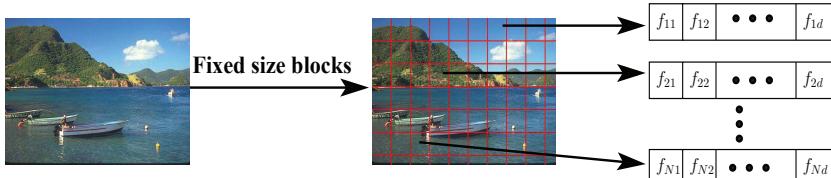


Fig. 13.5. Illustration of representing a scene image using local feature vectors that are extracted from fixed size non-overlapping blocks of the image.

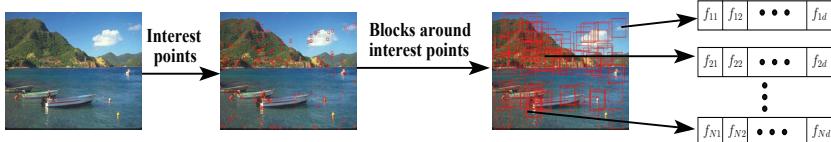


Fig. 13.6. Illustration of representing a scene image using local feature vectors that are extracted from fixed size blocks that are placed around interest points in the image.

Gaussian mixture models (GMMs) [36, 37] are commonly used for classification of varying length patterns represented as sets of local feature vectors. Hidden Markov models (HMMs) are commonly used for classification of varying length patterns represented as sequences of local feature vectors. The ML based method is commonly used for estimation of parameters of a GMM or a HMM for each class. The discriminative model based approaches to classification of varying length patterns represented as either sets of local feature vectors or sequences of local feature vectors include the support vector machine (SVM) based approaches [38]. There are two approaches to varying length pattern classification using SVMs depending on the type of kernel used. In the first approach, a varying length pattern is

first mapped onto a fixed length pattern and then a kernel for fixed length patterns is used to build the SVM [39–41]. In the second approach, a suitable kernel for varying length patterns is designed. The kernels designed for varying length patterns are called dynamic kernels [42]. The focus of this research work is to use dynamic kernel based SVMs for the analysis of varying length patterns of speech and image represented as sets of local feature vectors or sequences of local feature vectors. In the next section we present pattern classification using support vector machines (SVMs).

13.3. Support vector machines for pattern classification

In this section we describe the support vector machines (SVMs) for pattern classification. The SVM [38, 43, 44] is a linear two-class classifier. An SVM constructs the maximum margin hyperplane (optimal separating hyperplane) as a decision surface to separate the data points of two classes. The margin of a hyperplane is defined as the minimum distance of training points from the hyperplane. We first discuss the construction of an optimal separating hyperplane for linearly separable classes [45]. Then we discuss the construction of an optimal separating hyperplane for linearly nonseparable classes, *i.e.*, some training examples of the classes cannot be classified correctly. Finally, we discuss building an SVM for nonlinearly separable classes by constructing an optimal separating hyperplane in a high dimensional feature space corresponding to a nonlinear transformation induced by a kernel function.

13.3.1. Optimal separating hyperplane for linearly separable classes

Suppose the training data set consists of L examples, $\{(\mathbf{x}_i, y_i)\}_{i=1}^L$, $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{+1, -1\}$, where \mathbf{x}_i is i th training example and y_i is the corresponding class label. Figure 13.7 illustrates the construction of an optimal separating hyperplane for linearly separable classes in the two-dimensional input space of \mathbf{x} . A hyperplane is specified as $\mathbf{w}^\top \mathbf{x} + b = 0$, where \mathbf{w} is the parameter vector and b is the bias. The examples with class label $y_i = +1$ are the data points lying on the positive side of the hyperplane and the examples with class label $y_i = -1$ are the data points lying on the negative side of the hyperplane. Now, a separating hyperplane that separates the data points of two linearly separable classes satisfies the

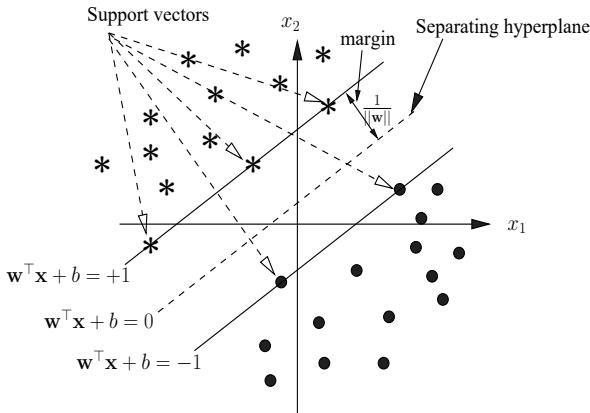


Fig. 13.7. Illustration of constructing the optimal separating hyperplane for linearly separable classes.

following constraints:

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) > 0 \quad \text{for } i = 1, 2, \dots, L \quad (13.1)$$

The distance between the nearest example and the separating hyperplane, called the margin, is given by $\frac{1}{\|\mathbf{w}\|}$. The problem of finding the optimal separating hyperplane that maximizes the margin is the same as the problem of minimizing the Euclidean norm of the parameter vector \mathbf{w} . For reducing the search space of \mathbf{w} , the constraints that the optimal separating hyperplane must satisfy are specified as follows:

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \quad \text{for } i = 1, 2, \dots, L \quad (13.2)$$

The learning problem of finding the optimal separating hyperplane is a constrained optimization problem stated as follows: Given the training data set, find the values of \mathbf{w} and b such that they satisfy the constraints in (13.2) and the parameter vector \mathbf{w} minimizes the following cost function:

$$\mathcal{J}(\mathbf{w}) = \frac{1}{2} \|\mathbf{w}\|^2 \quad (13.3)$$

The constrained optimization problem is solved using the method of Lagrangian multipliers. The primal form of the Lagrangian objective function is given by

$$\mathcal{L}_p(\mathbf{w}, b, \alpha) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^L \alpha_i [y_i(\mathbf{w}^\top \mathbf{x}_i + b) - 1] \quad (13.4)$$

where the non-negative variables α_i are called Lagrange multipliers. The saddle point of the Lagrangian objective function provides the solution for

the optimization problem. The solution is determined by first minimizing the Lagrangian objective function with respect to \mathbf{w} and b , and then maximizing with respect to $\boldsymbol{\alpha}$. The two conditions of optimality due to minimization are

$$\frac{\partial \mathcal{L}_p(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial \mathbf{w}} = \mathbf{0} \quad (13.5)$$

$$\frac{\partial \mathcal{L}_p(\mathbf{w}, b, \boldsymbol{\alpha})}{\partial b} = 0 \quad (13.6)$$

Application of optimality conditions gives

$$\mathbf{w} = \sum_{i=1}^L \alpha_i y_i \mathbf{x}_i \quad (13.7)$$

$$\sum_{i=1}^L \alpha_i y_i = 0 \quad (13.8)$$

Substituting the expression for \mathbf{w} from (13.7) in (13.4) we get

$$\mathcal{L}_d(\boldsymbol{\alpha}) = \frac{1}{2} \left(\sum_{i=1}^L \alpha_i y_i \mathbf{x}_i \right)^\top \left(\sum_{j=1}^L \alpha_j y_j \mathbf{x}_j \right) - \sum_{i=1}^L \alpha_i \left[y_i \left(\sum_{j=1}^L \alpha_j y_j \mathbf{x}_j^\top \right) \mathbf{x}_i + b - 1 \right] \quad (13.9)$$

$$\begin{aligned} \mathcal{L}_d(\boldsymbol{\alpha}) &= \frac{1}{2} \left(\sum_{i=1}^L \alpha_i y_i \mathbf{x}_i \right)^\top \left(\sum_{j=1}^L \alpha_j y_j \mathbf{x}_j \right) \\ &\quad - \sum_{i=1}^L \sum_{j=1}^L \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j - \sum_{i=1}^L \alpha_i y_i b + \sum_{i=1}^L \alpha_i \end{aligned} \quad (13.10)$$

Using the condition in (13.8) and simplifying (13.10), the dual form of Lagrangian objective function can be derived as a function of Lagrangian multipliers $\boldsymbol{\alpha}$, as follows:

$$\mathcal{L}_d(\boldsymbol{\alpha}) = \sum_{i=1}^L \alpha_i - \frac{1}{2} \sum_{i=1}^L \sum_{j=1}^L \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j \quad (13.11)$$

The optimal values of Lagrangian multipliers are determined by maximizing the objective function $L_d(\boldsymbol{\alpha})$ subject to the following constraints:

$$\sum_{i=1}^L \alpha_i y_i = 0 \quad (13.12)$$

$$\alpha_i \geq 0 \quad \text{for } i = 1, 2, \dots, L \quad (13.13)$$

This optimization problem is solved using quadratic programming methods [46]. The data points for which the values of the optimum Lagrange

multipliers are not zero are the support vectors. For these data points the distance to the optimal separating hyperplane is minimum. Hence, the support vectors are the training data points that lie on the margin, as illustrated in Figure 13.7. Support vectors are those data points that lie closest to the decision surface and define the optimum parameter vector \mathbf{w} . For the optimum Lagrange multipliers $\{\alpha_j^*\}_{j=1}^{L_s}$, the optimum parameter vector \mathbf{w}^* is given by

$$\mathbf{w}^* = \sum_{j=1}^{L_s} \alpha_j^* y_j \mathbf{x}_j \quad (13.14)$$

where L_s is the number of support vectors. The discriminant function of the optimal separating hyperplane in terms of support vectors is given by

$$D(\mathbf{x}) = \mathbf{w}^{*\top} \mathbf{x} + b^* = \sum_{j=1}^{L_s} \alpha_j^* y_j \mathbf{x}^\top \mathbf{x}_j + b^* \quad (13.15)$$

where b^* is the optimum bias.

However, the data for most of the real world tasks are not linearly separable. Next we present a method to construct an optimal separating hyperplane for linearly non-separable classes.

13.3.2. Optimal separating hyperplane for linearly non-separable classes

The training data points of the linearly non-separable classes cannot be separated by a hyperplane without classification error. In such cases, it is desirable to find an optimal separating hyperplane that minimizes the probability of classification error for the training data set. A data point is non-separable when it does not satisfy the constraint in (13.2). This corresponds to a data point that falls either within margin or on the wrong side of the separating hyperplane as illustrated in Figure 13.8.

For linearly non-separable classes, the constraints in (13.2) are modified by introducing the non-negative slack variables ξ_i as follows:

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i \quad \text{for } i = 1, 2, \dots, L \quad (13.16)$$

The slack variable ξ_i is a measure of the deviation of a data point \mathbf{x}_i from the ideal condition of separability. For $0 \leq \xi_i \leq 1$, the data point falls inside the region of separation, but on the correct side of the separating hyperplane. For $\xi_i > 1$, the data point falls on the wrong side of the separating hyperplane. The support vectors are those particular data points that satisfy the constraint in (13.16) with equality sign. The cost function for linearly non-separable classes is given as

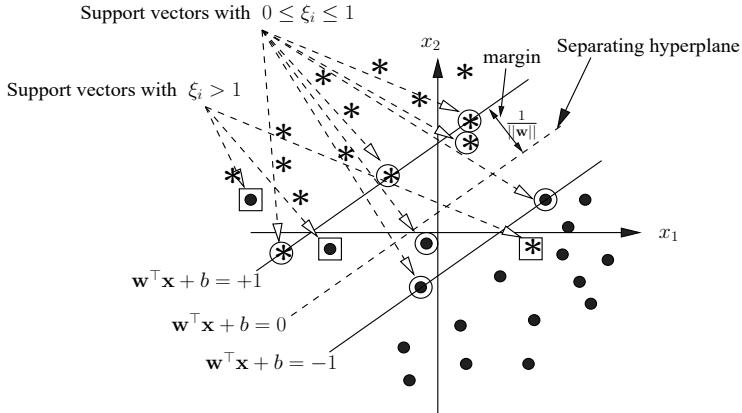


Fig. 13.8. Illustration of constructing the optimal separating hyperplane for linearly non-separable classes. Support vectors covered with circle correspond to $0 \leq \xi_i \leq 1$ and support vectors covered with square correspond to $\xi_i > 1$.

$$J(\mathbf{w}, \boldsymbol{\xi}) = \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^L \xi_i \quad (13.17)$$

where C is a user-specified positive parameter that controls the trade-off between the complexity of the classifier and the number of non-separable data points. Using the method of Lagrange multipliers to solve the constrained optimization problem as in the case of linearly separable classes, the dual form of the Lagrangian objective function can be obtained as follows [45]:

$$\mathcal{L}_d(\boldsymbol{\alpha}) = \sum_{i=1}^L \alpha_i - \frac{1}{2} \sum_{i=1}^L \sum_{j=1}^L \alpha_i \alpha_j y_i y_j \mathbf{x}_i^\top \mathbf{x}_j \quad (13.18)$$

subject to the constraints:

$$\sum_{i=1}^L \alpha_i y_i = 0 \quad (13.19)$$

$$0 \leq \alpha_i \leq C \quad \text{for } i = 1, 2, \dots, L \quad (13.20)$$

It may be noted that the maximum value that the Lagrangian multipliers $\{\alpha_i\}$ can take is C for the linearly non-separable classes. For the optimum Lagrange multipliers $\{\alpha_j^*\}_{j=1}^{L_s}$, the optimum parameter vector \mathbf{w}^* is given by

$$\mathbf{w}^* = \sum_{j=1}^{L_s} \alpha_j^* y_j \mathbf{x}_j \quad (13.21)$$

where L_s is the number of support vectors. The discriminant function of the optimal separating hyperplane for an input vector \mathbf{x} is given by

$$D(\mathbf{x}) = \mathbf{w}^* \top \mathbf{x} + b^* = \sum_{j=1}^{L_s} \alpha_j^* y_j \mathbf{x}_j \top \mathbf{x}_j + b^* \quad (13.22)$$

where b^* is the optimum bias.

13.3.3. Support vector machine for nonlinearly separable classes

For nonlinearly separable classes, an SVM is built by mapping the input vector $\mathbf{x}_i, i = 1, 2, \dots, L$ into a high dimensional feature vector $\Phi(\mathbf{x}_i)$ using a nonlinear transformation Φ , and constructing an optimal separating hyperplane defined by $\mathbf{w}^\top \Phi(\mathbf{x}) + b = 0$ to separate the examples of two classes in the feature space $\Phi(\mathbf{x})$. This is based on Cover's theorem [47] which states that an input space where the patterns are nonlinearly separable may be transformed into a feature space where the patterns are linearly separable with a high probability, provided two conditions are satisfied [45]. The first condition is that the transformation is nonlinear and the second condition is that the dimensionality of the feature space is high enough. The concept of nonlinear transformation used in building an SVM for nonlinearly separable classes is illustrated in Figure 13.9. It is seen that the nonlinearly separable data points $\mathbf{x}_i = [x_{i1}, x_{i2}]^\top, i = 1, 2, \dots, L$ in a two-dimensional input space are mapped onto three-dimensional feature vectors $\Phi(\mathbf{x}_i) = [x_{i1}^2, x_{i2}^2, \sqrt{2}x_{i1}x_{i2}]^\top, i = 1, 2, \dots, L$ where they are linearly separable.

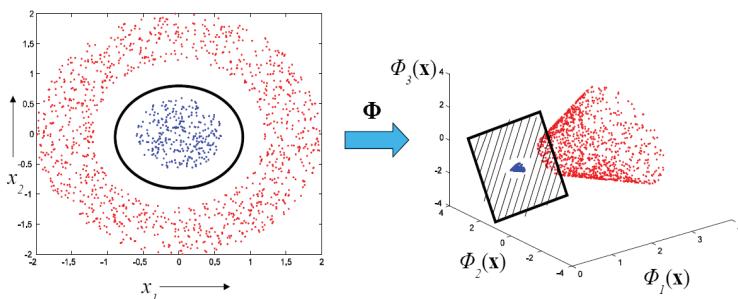


Fig. 13.9. Illustration of nonlinear transformation used in building an SVM for nonlinearly separable classes.

For the construction of the optimal separating hyperplane in the high dimensional feature space $\Phi(\mathbf{x})$, the dual form of the Lagrangian objective function in (13.18) takes the following form:

$$\mathcal{L}_d(\boldsymbol{\alpha}) = \sum_{i=1}^L \alpha_i - \frac{1}{2} \sum_{i=1}^L \sum_{j=1}^L \alpha_i \alpha_j y_i y_j \Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}_j) \quad (13.23)$$

subject to the constraints:

$$\sum_{i=1}^L \alpha_i y_i = 0 \quad (13.24)$$

and

$$0 \leq \alpha_i \leq C \quad \text{for } i = 1, 2, \dots, L \quad (13.25)$$

For the optimal $\boldsymbol{\alpha}^*$, the optimal parameter vector \mathbf{w}^* is given by

$$\mathbf{w}^* = \sum_{j=1}^{L_s} \alpha_j^* y_j \Phi(\mathbf{x}_j) \quad (13.26)$$

where L_s is the number of support vectors. The discriminant function of the optimal separating hyperplane for an input vector \mathbf{x} is defined as

$$D(\mathbf{x}) = \mathbf{w}^{*\top} \Phi(\mathbf{x}) + b^* = \sum_{j=1}^{L_s} \alpha_j^* y_j \Phi(\mathbf{x})^\top \Phi(\mathbf{x}_j) + b^* \quad (13.27)$$

Solving (13.23) involves computation of the innerproduct operation $\Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}_j)$. Evaluation of innerproducts in a high dimensional feature space is avoided by using an innerproduct kernel, $K(\mathbf{x}_i, \mathbf{x}_j)$, defined as $K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i)^\top \Phi(\mathbf{x}_j)$ [48]. A valid innerproduct kernel $K(\mathbf{x}_i, \mathbf{x}_j)$ for two pattern vectors \mathbf{x}_i and \mathbf{x}_j is a symmetric function for which the Mercer's theorem holds good. The Mercer's theorem can be stated as follows [45]. Let $K(\mathbf{x}_i, \mathbf{x}_j)$ be a continuous symmetric kernel that is defined in the closed interval $u \leq \mathbf{x}_i \leq v$ and likewise for \mathbf{x}_j . The kernel $K(\mathbf{x}_i, \mathbf{x}_j)$ can be expanded in the series

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{l=1}^{\infty} \lambda_l \varphi_l(\mathbf{x}_i) \varphi_l(\mathbf{x}_j) \quad (13.28)$$

with positive coefficients, $\lambda_l > 0$ for all l . For this expansion to be valid and for it to be converge absolutely and uniformly, it is necessary and sufficient that the condition

$$\int_u^v \int_u^v K(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_i) g(\mathbf{x}_j) d\mathbf{x}_i d\mathbf{x}_j \geq 0 \quad (13.29)$$

holds for all $g(\mathbf{x}_i)$ such that

$$\int_u^v g^2(\mathbf{x}_i) d\mathbf{x}_i < \infty \quad (13.30)$$

The functions $\varphi_l(\mathbf{x}_i)$ are called eigen functions of the expansion and the numbers λ_l are called eigenvalues. The fact that all of the eigenvalues are positive means that the kernel $K(\mathbf{x}_i, \mathbf{x}_j)$ is positive definite.

The objective function in (13.23) and the discriminant function of the optimal separating hyperplane in (13.27) can now be specified using the kernel function as follows:

$$\mathcal{L}_d(\boldsymbol{\alpha}) = \sum_{i=1}^L \alpha_i - \frac{1}{2} \sum_{i=1}^L \sum_{j=1}^L \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \quad (13.31)$$

$$D(\mathbf{x}) = \mathbf{w}^{*\top} \Phi(\mathbf{x}) + b^* = \sum_{j=1}^{L_s} \alpha_j^* y_j K(\mathbf{x}, \mathbf{x}_j) + b^* \quad (13.32)$$

The architecture of a support vector machine for two-class pattern classification that implements the discriminant function of the hyperplane in (13.32) is given in Figure 13.10. The number of hidden nodes corresponding to the number of support vectors, and the training examples corresponding

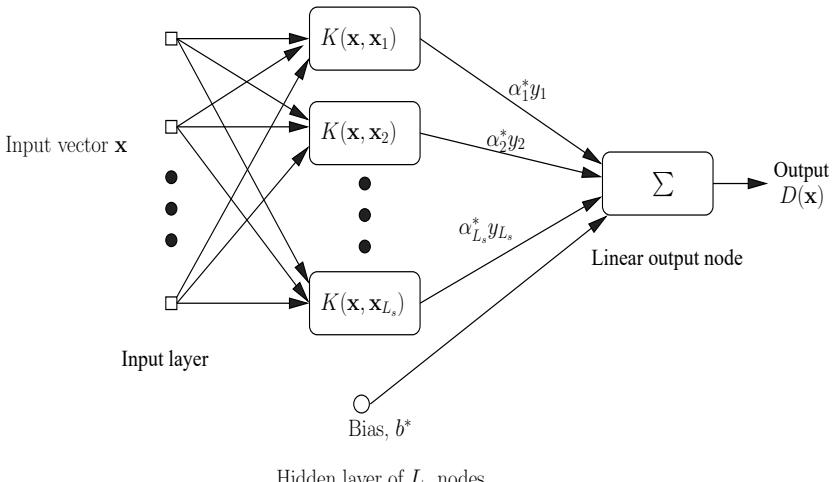


Fig. 13.10. Architecture of a support vector machine for two-class pattern classification. The class of the input pattern \mathbf{x} is given by the sign of the discriminant function $D(\mathbf{x})$. The number of hidden nodes corresponds to the number of support vectors L_s . Each hidden node computes the innerproduct kernel function $K(\mathbf{x}, \mathbf{x}_i)$ on the input pattern \mathbf{x} and a support vector \mathbf{x}_i .

to the support vectors are determined by maximizing the objective function in (13.31) using a given training data set and for a chosen kernel function.

Some commonly used innerproduct kernel functions are as follows:

$$\text{Polynomial kernel: } K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{a}\mathbf{x}_i^\top \mathbf{x}_j + c)^p$$

$$\text{Sigmoidal kernel: } K(\mathbf{x}_i, \mathbf{x}_j) = \tanh(\mathbf{a}\mathbf{x}_i^\top \mathbf{x}_j + c)$$

$$\text{Gaussian kernel: } K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\delta\|\mathbf{x}_i - \mathbf{x}_j\|^2)$$

Here, \mathbf{x}_i and \mathbf{x}_j are vectors in the d -dimensional input pattern space, a and c are constants, p is the degree of the polynomial and δ is a non-negative constant used for numerical stability in Gaussian kernel function. The dimensionality of the feature space is $\binom{p+d}{d}$ for the polynomial kernel [43]. The feature spaces for the sigmoidal and Gaussian kernels are of infinite dimension. The kernel functions involve computations in the d -dimensional input space and avoid the innerproduct operations in the high dimensional feature space. The above mentioned kernel functions are kernels for vectorial data. The kernels for non-vectorial data include kernel on graph, kernels on sets, kernels on text, kernels on structured data like string, trees etc [49]. The graph kernel [50] include random walk kernels, shortest paths kernel, and all paths kernel are used in bioinformatics for protein function prediction, comparing structures of proteins, comparing structures of RNA, and biological network comparison. The kernels on sets include intersection kernel, union complement kernel, and agreement kernel. Kernels on text include vector space kernels, semantic kernels, and latent semantic kernels [49]. These kernels are used for text categorization and document classification. Spectrum kernel [51], mismatch kernel [52, 53], string subsequence kernel [54], and marginalised kernels [55] are the examples of string kernels. These kernels are used in various applications in bioinformatics such as protein classification, protein function prediction, and protein structure prediction. The best choice of the kernel function for a given pattern classification problem is still a research issue [38]. The suitable kernel function and its parameters are chosen empirically.

The complexity of a two class support vector machine is a function of the number of support vectors (L_s) determined during its training. Multiclass pattern classification problems are generally solved using a combination of two-class SVMs. Therefore, the complexity of a multiclass pattern classification system depends on the number of SVMs and the complexity of each SVM used. In the next subsection, we present the commonly used approaches to multiclass pattern classification using SVMs.

13.3.4. Multi-class pattern classification using SVMs

Support vector machines are originally designed for two-class pattern classification. Multi-class pattern classification problems are commonly solved using a combination of two-class SVMs and a decision strategy to decide the class of the input pattern [56]. Each SVM has the architecture given in Figure 13.10 and is trained independently. Now we present the two approaches to decomposition of the learning problem in multi-class pattern classification into several two-class learning problems so that a combination of SVMs can be used. The training data set $\{(\mathbf{x}_i, l_i)\}$ consists of L examples belonging to C classes. The class label is $l_i \in \{1, 2, \dots, C\}$. For the sake of simplicity, we assume that the number of examples for each class is the same, i.e., L/C .

13.3.4.1. One-against-the-rest approach

In this approach, an SVM is constructed for each class by discriminating that class against the remaining ($C-1$) classes. The classification system based on this approach consists of C SVMs. All the L training examples are used in constructing an SVM for each class. In constructing the SVM for the class c the desired output y_i for a training example \mathbf{x}_i is specified as follows:

$$\begin{aligned} y_i &= +1, && \text{if } l_i = c \\ &= -1, && \text{if } l_i \neq c \end{aligned} \quad (13.33)$$

The examples with the desired output $y_i = +1$ are called *positive* examples. The examples with the desired output $y_i = -1$ are called *negative* examples. An optimal separating hyperplane is constructed to separate L/C positive examples from $L(C-1)/C$ negative examples. The much larger number of negative examples leads to an imbalance, resulting in the dominance of negative examples in determining the decision boundary [57]. The extent of imbalance increases with the number of classes and is significantly high when the number of classes is large. A test pattern \mathbf{x} is classified by using the *winner-takes-all* strategy that uses the following decision rule:

$$\text{Class label for } \mathbf{x} = \arg \max_c D_c(\mathbf{x}) \quad (13.34)$$

where $D_c(\mathbf{x})$ is the discriminant function of the SVM constructed for the class c .

13.3.4.2. One-against-one approach

In this approach, an SVM is constructed for every pair of classes by training it to discriminate the two classes. The number of SVMs used in this

approach is $C(C-1)/2$. An SVM for a pair of classes c and s is constructed using $2L/C$ training examples belonging to the two classes only. The desired output y_i for a training example \mathbf{x}_i is specified as follows:

$$\begin{aligned} y_i &= +1, & \text{if } l_i = c \\ &= -1, & \text{if } l_i = s \end{aligned} \quad (13.35)$$

The small size of the set of training examples and the balance between the number of positive and negative examples lead to a simple optimization problem to be solved in constructing an SVM for a pair of classes. When the number of classes is large, the proliferation of SVMs leads to a complex classification system.

The *maxwins* strategy is commonly used to determine the class of a test pattern \mathbf{x} in this approach. In this strategy, a majority voting scheme is used. If $D_{cs}(\mathbf{x})$, the value of the discriminant function of the SVM for the pair of classes c and s , is positive, then the class c wins a vote. Otherwise, the class s wins a vote. Outputs of SVMs are used to determine the number of votes won by each class. The class with the maximum number of votes is assigned to the test pattern. When there are multiple classes with the same maximum number of votes, the class with the maximum value of the total magnitude of discriminant functions (TMDF) is assigned. The total magnitude of discriminant functions for the class c is defined as follows:

$$\text{TMDF}_c = \sum_s |D_{cs}(\mathbf{x})| \quad (13.36)$$

where the summation is over all s with which the class c is paired. The maxwins strategy needs evaluation of discriminant functions of all the SVMs in deciding the class of a test pattern.

It is necessary to use suitable kernels for pattern classification using SVMs. The commonly used kernels such as Gaussian kernel and polynomial kernel are not suitable for SVM-based classification of sets or sequences of local feature vectors representation of varying length patterns. Kernels suitable for sets or sequences of local feature vectors representation of varying length patterns are known as dynamic kernels. In the next section, we present the issues in designing dynamic kernels for varying length patterns.

13.4. Design of kernels for varying length patterns

Classification of varying length patterns of speech requires that the speech signal of an utterance is processed using a short-time analysis technique to represent it as a set or a sequence of local feature vectors. For the tasks

in which sequence information is not considered to be critical, the set of local feature vectors representation is considered. The sequence of local feature vectors representation is considered for the tasks that need to use the sequence information. The size of the set or the length of the sequence of local feature vectors depends on the duration of the utterance. Support vector machine (SVM) using a kernel such as Gaussian kernel and polynomial kernel can not handle varying length patterns. The kernels designed for varying length patterns are referred to as dynamic kernels [42]. Different approaches to design dynamic kernels are as follows: (1) Explicit mapping based approaches [3, 58] that involve mapping a set of local feature vectors or a sequence of local feature vectors onto a fixed dimensional representation and then defining a kernel function in the space of that representation; (2) Probabilistic distance metric based approaches [4, 59] that involve kernelizing a suitable distance measure between the probability distributions corresponding to the two sets or sequences of local feature vectors; (3) Matching based approaches [5] that involve computing a kernel function by matching the local feature vectors in the pair of sets or sequences of local feature vectors; and (4) Dynamic time alignment based approaches [60, 61] that involve defining a kernel function using the dynamic time warping (DTW) distance [8] between a pair of sequences of local feature vectors. The Fisher kernel using GMM-based likelihood score vectors [62] and probabilistic sequence kernel [58] are the dynamic kernels for sets of local feature vectors constructed using the explicit mapping based approaches. The Fisher kernel using HMM-based likelihood score vectors [3] and likelihood ratio kernel [63] are the dynamic kernels for sequences of local feature vectors constructed using the explicit mapping based approaches. The GMM supervector kernel [4] and GMM universal background model (UBM) mean interval kernel [59], the state-of-the-art dynamic kernels for sets of local feature vectors, are designed using the probabilistic distance metric based approaches. The probability product kernel [64] is the dynamic kernel for the sequences of local feature vectors designed using probabilistic distance metric based approaches. The summation kernel [65], matching kernel [66] and intermediate matching kernel [5] are the dynamic kernels for sets of local feature vectors designed using the matching based approaches. The DTW kernel [67], dynamic time alignment kernel [60], global alignment kernel [68] and triangular alignment kernel [61] are the dynamic kernels for sequences of local feature vectors designed using the alignment based approaches. In the following sections, we first describe the dynamic kernels for sets of local feature vectors and then we describe the dynamic kernels for sequences of local feature vectors.

13.5. Dynamic kernels for sets of local feature vectors

In this section, we present the Fisher kernel, probabilistic sequence kernel, GMM supervector kernel, GMM-UBM mean interval kernel, intermediate matching kernel (IMK) and pyramid matching kernel (PMK) for sets of local feature vectors.

13.5.1. Fisher kernel for sets of local feature vectors

The Fisher kernel (FK) [62] for sets of local feature vectors uses an explicit expansion into a kernel feature space defined by a GMM-based likelihood score space in which a set of local feature vectors is represented as a fixed dimensional Fisher score vector. Likelihood score space is formed using the first order derivatives of the log-likelihood with respect to the parameters of a GMM. For a set of d -dimensional local feature vectors, $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$, the first order derivative of the log-likelihood, i.e., the gradient vector of the log-likelihood, with respect to mean vector of the q th component of the GMM, $\boldsymbol{\mu}_q$, is given by

$$\varphi_q^{(\mu)}(\mathbf{X}) = \sum_{t=1}^T \gamma_q(\mathbf{x}_t) \mathbf{z}_{tq} \quad (13.37)$$

where $\mathbf{z}_{tq} = \Sigma_q^{-1}(\mathbf{x}_t - \boldsymbol{\mu}_q)$ and $\gamma_q(\mathbf{x}_t)$, the responsibility of the component q for the local feature vector \mathbf{x}_t , is given by

$$\gamma_q(\mathbf{x}_t) = \frac{w_q \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)}{\sum_{q'=1}^Q w_{q'} \mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_{q'}, \boldsymbol{\Sigma}_{q'})} \quad (13.38)$$

Here Q is the number of components in the GMM, w_q is the mixture coefficient of the component q , and $\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$ is the normal density for the component q with mean vector $\boldsymbol{\mu}_q$ and covariance matrix $\boldsymbol{\Sigma}_q$. The gradient vector of the log-likelihood with respect to $\boldsymbol{\Sigma}_q$ is given by

$$\varphi_q^{(\Sigma)}(\mathbf{X}) = \frac{1}{2} \sum_{t=1}^T \gamma_q(\mathbf{x}_t) \left[-\mathbf{u}_q + \mathbf{v}_{tq} \right] \quad (13.39)$$

where $\mathbf{u}_q = [\sigma_{11}^{(q)}, \sigma_{12}^{(q)}, \dots, \sigma_{dd}^{(q)}]^T$ and $\mathbf{v}_{tq} = [z_{1q} \mathbf{z}_{tq}^\top, z_{2q} \mathbf{z}_{tq}^\top, \dots, z_{dq} \mathbf{z}_{tq}^\top]^T$. Here, \mathbf{u}_q and \mathbf{v}_{tq} are d^2 -dimensional vectors. The gradient of the log-likelihood with respect to w_q is given by

$$\varphi_q^{(w)}(\mathbf{X}) = \sum_{t=1}^T \gamma_q(\mathbf{x}_t) \left[\frac{1}{w_q} - \frac{\gamma_1(\mathbf{x}_t)}{w_1 \gamma_q(\mathbf{x}_t)} \right] \quad (13.40)$$

The Fisher score vector with respect to the parameters of the q th component of the GMM is obtained as a supervector of gradient vectors of the log-likelihood for that component and is given by

$$\hat{\Phi}_q(\mathbf{X}) = \left[\varphi_q^{(\mu)}(\mathbf{X})^\top, \varphi_q^{(\Sigma)}(\mathbf{X})^\top, \varphi_q^{(w)}(\mathbf{X})^\top \right]^\top \quad (13.41)$$

Now, a set of local feature vectors \mathbf{X} is represented as a fixed dimensional supervector $\Phi_{\text{FK}}(\mathbf{X})$ of all the Q Fisher score vectors as follows:

$$\Phi_{\text{FK}}(\mathbf{X}) = \left[\hat{\Phi}_1(\mathbf{X})^\top, \hat{\Phi}_2(\mathbf{X})^\top, \dots, \hat{\Phi}_Q(\mathbf{X})^\top \right]^\top \quad (13.42)$$

The dimension of Fisher score vector is $D=Q \times (d + d^2 + 1)$. The Fisher kernel between two sets of local feature vectors \mathbf{X}_m and \mathbf{X}_n is computed as

$$K_{\text{FK}}(\mathbf{X}_m, \mathbf{X}_n) = \Phi_{\text{FK}}(\mathbf{X}_m)^\top \mathbf{F}^{-1} \Phi_{\text{FK}}(\mathbf{X}_n) \quad (13.43)$$

Here \mathbf{F} is the Fisher information matrix given as

$$\mathbf{F} = \frac{1}{L} \sum_{l=1}^L \Phi_{\text{FK}}(\mathbf{X}_l) \Phi_{\text{FK}}(\mathbf{X}_l)^\top \quad (13.44)$$

where L is the number of training examples. Fisher kernel is a class-specific kernel, as it uses the GMM of a class to compute the kernel for that class. Hence the SVM-based classifier with Fisher kernel uses the one-against-the-rest approach to multi-class pattern classification.

The computation of each of the three gradient vectors involves $Q \times (T_m + T_n)$ operations. The computation of Fisher information matrix involves $L \times D^2 + L$ operations and kernel computation involves $D^2 + D$ operations. Hence the computational complexity of Fisher kernel is $\mathcal{O}(QT + LD^2 + L + D^2 + D)$, where T is the maximum of set cardinalities T_m and T_n . However, the computation of Fisher information matrix and its inverse is computationally intensive. For a 128-component GMM that uses diagonal covariance matrix on the 39-dimensional feature vectors, the dimension of the resulting supervector of Fisher score vectors is 10112, and the dimension of the Fisher information matrix is 10112×10112 .

13.5.2. Probabilistic sequence kernel

Probabilistic sequence kernel (PSK) [58] maps a set of local feature vectors onto a probabilistic feature vector obtained using generative models. The PSK uses the universal background model (UBM) with Q mixtures [69] and the class-specific GMMs obtained by adapting the UBM. The likelihood of

a local feature vector \mathbf{x} being generated by the $2Q$ -mixture GMM that includes the UBM and a class-specific GMM is given as

$$p(\mathbf{x}) = \sum_{q=1}^{2Q} p(\mathbf{x}|q)P(q) \quad (13.45)$$

where $P(q) = w_q$ and $p(\mathbf{x}|q) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$. The normalized Gaussian basis function for the q th component is defined as

$$\varphi_q(\mathbf{x}) = \frac{p(\mathbf{x}|q)P(q)}{\sum_{q'=1}^{2Q} p(\mathbf{x}|q')P(q')} \quad (13.46)$$

A local feature vector \mathbf{x} is represented in a higher dimensional feature space as a vector of normalized Gaussian basis functions, $\boldsymbol{\varphi}(\mathbf{x}) = [\varphi_1(\mathbf{x}), \varphi_2(\mathbf{x}), \dots, \varphi_{2Q}(\mathbf{x})]^\top$. Since the element $\varphi_q(\mathbf{x})$ indicates the probabilistic alignment of \mathbf{x} to the q th component, $\boldsymbol{\varphi}(\mathbf{x})$ is called the probabilistic alignment vector. A set of local feature vectors $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$ is represented as a fixed dimensional vector $\Phi_{PSK}(\mathbf{X})$ in the higher dimensional space, as given by

$$\Phi_{PSK}(\mathbf{X}) = \frac{1}{T} \sum_{t=1}^T \boldsymbol{\varphi}(\mathbf{x}_t) \quad (13.47)$$

The dimension of $\Phi_{PSK}(\mathbf{X})$ is $D=2Q$. The PSK between two examples $\mathbf{X}_m = \{\mathbf{x}_{m1}, \mathbf{x}_{m2}, \dots, \mathbf{x}_{mT_m}\}$ and $\mathbf{X}_n = \{\mathbf{x}_{n1}, \mathbf{x}_{n2}, \dots, \mathbf{x}_{nT_n}\}$ is given as

$$K_{PSK}(\mathbf{X}_m, \mathbf{X}_n) = \Phi_{PSK}(\mathbf{X}_m)^\top \mathbf{S}^{-1} \Phi_{PSK}(\mathbf{X}_n) \quad (13.48)$$

The correlation matrix \mathbf{S} is defined as follows:

$$\mathbf{S} = \frac{1}{M} \mathbf{R}^\top \mathbf{R} \quad (13.49)$$

where \mathbf{R} is the matrix whose rows are the probabilistic alignment vectors for local feature vectors of examples in the training data set and M is the total number of local feature vectors in the training data set. The PSK is a class-specific kernel, as it uses the GMM for a class to compute the kernel for that class. Hence the SVM classifier with PSK uses the one-against-the-rest approach to multi-class pattern classification.

The computation of the fixed dimensional representation involves $2Q \times (T_m + T_n)$ operations. The computation of correlation matrix involves $M \times D^2 + M$ operations and kernel computation involves $D^2 + D$ operations. Hence the computational complexity of PSK is $\mathcal{O}(QT + MD^2 + M + D^2 + D)$, where T is the maximum of set cardinalities T_m and T_n .

13.5.3. GMM supervector kernel

The GMM supervector (GMMSV) kernel [4] performs a mapping of a set of local feature vectors onto a higher dimensional vector corresponding to a GMM supervector. An UBM with Q components is built using the training examples of all the classes. An example-specific GMM is built for each example by adapting the means of the UBM using the data of that example. Let $\mu_q^{(\mathbf{X})}$ be the mean vector of the q th component in the example-specific GMM for an example $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$. A GMM vector $\varphi_q(\mathbf{X})$ for an example \mathbf{X} corresponding to the q th component of GMM is obtained as follows:

$$\varphi_q(\mathbf{X}) = \left[\sqrt{w_q} \Sigma_q^{-\frac{1}{2}} \mu_q^{(\mathbf{X})} \right] \quad (13.50)$$

The GMM supervector for the example \mathbf{X} is given by

$$\Phi_{\text{GMMSV}}(\mathbf{X}) = [\varphi_1(\mathbf{X})^\top, \varphi_2(\mathbf{X})^\top, \dots, \varphi_Q(\mathbf{X})^\top]^\top \quad (13.51)$$

The dimension of GMM supervector is $D=Q \times d$, where d is the dimension of local feature vectors. The GMMSV kernel between a pair of examples \mathbf{X}_m and \mathbf{X}_n is given by

$$K_{\text{GMMSVK}}(\mathbf{X}_m, \mathbf{X}_n) = \Phi_{\text{GMMSV}}(\mathbf{X}_m)^\top \Phi_{\text{GMMSV}}(\mathbf{X}_n) \quad (13.52)$$

It is shown in [4] that the resulting kernel value is equivalent to the divergence between two example-specific GMMs using Kullback-Leibler divergence. The GMMSV kernel is computed using a total of $Q \times (T_m + T_n)$ computations in mean adaptation and $Q \times (d^2 + 1)$ computations in generating the GMM supervector and D^2 operations in kernel computation. Hence the computational complexity of GMMSV kernel is $\mathcal{O}(QT + Qd^2 + Q + D^2)$, where T is the maximum of set cardinalities T_m and T_n .

13.5.4. GMM-UBM mean interval kernel

The construction of GMM-UBM mean interval (GUMI) kernel [59] also involves building the UBM. An example-specific GMM is built for each example by adapting the mean vectors and covariance matrices of the UBM using the data of that example. Let $\mu_q^{(\mathbf{X})}$ and $\Sigma_q^{(\mathbf{X})}$ be the mean vector and the covariance matrix of q th component in the example-specific GMM for an example $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T\}$. A GUMI vector $\varphi_q(\mathbf{X})$ for an example \mathbf{X} corresponding to the q th component of GMM is obtained as follows:

$$\varphi_q(\mathbf{X}) = \left(\frac{\Sigma_q^{(\mathbf{X})} + \Sigma_q}{2} \right)^{-\frac{1}{2}} (\mu_q^{(\mathbf{X})} - \mu_q) \quad (13.53)$$

The GUMI supervector is obtained by concatenating the GUMI vectors of different components as

$$\Phi_{\text{GUMI}}(\mathbf{X}) = [\varphi_1(\mathbf{X})^\top, \varphi_2(\mathbf{X})^\top, \dots, \varphi_Q(\mathbf{X})^\top]^\top \quad (13.54)$$

The dimension of GUMI supervector is $D=Q \times d$, where d is the dimension of local feature vectors. The GUMI kernel between a pair of examples \mathbf{X}_m and \mathbf{X}_n is given by

$$K_{\text{GUMIK}}(\mathbf{X}_m, \mathbf{X}_n) = \Phi_{\text{GUMI}}(\mathbf{X}_m)^\top \Phi_{\text{GUMI}}(\mathbf{X}_n) \quad (13.55)$$

It is shown in [59] that the resulting kernel value is equivalent to the divergence between two example-specific GMMs using Bhattacharyya distance. The GUMI kernel is computed using a total of $Q \times (T_m + T_n)$ computations in mean adaptation, $Q \times (T_m + T_n)$ computations in covariance adaptation, $Q \times (d^2 + d)$ computations in generating the GUMI supervector and D^2 operations in kernel computation. Hence the computational complexity of GUMI kernel is $\mathcal{O}(QT + Qd^2 + Qd + D^2)$, where T is the maximum of set cardinalities T_m and T_n .

13.5.5. Matching based kernel for sets of local feature vectors

In this section, we present the different matching based approaches to design the dynamic kernels for the varying length patterns of examples represented by sets of local feature vectors.

13.5.5.1. Summation kernel

Let $\mathbf{X}_m = \{\mathbf{x}_{m1}, \mathbf{x}_{m2}, \dots, \mathbf{x}_{mT_m}\}$ and $\mathbf{X}_n = \{\mathbf{x}_{n1}, \mathbf{x}_{n2}, \dots, \mathbf{x}_{nT_n}\}$ be the sets of local feature vectors for two examples. The summation kernel (SK) [65] is computed by matching every local feature vector in \mathbf{X}_m with every local feature vector in \mathbf{X}_n as follows:

$$K_{\text{SK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{t=1}^{T_m} \sum_{t'=1}^{T_n} k(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) \quad (13.56)$$

where $k(.,.)$ is a base kernel. It can be proved that the summation kernel is a Mercer kernel.

13.5.5.2. Matching kernel

The matching kernel (MK) [66] is constructed by considering the closest local feature vector of an example for each local feature vector in the other example as follows:

$$K_{\text{MK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{t=1}^{T_m} \max_{t'} k(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) + \sum_{t'=1}^{T_n} \max_t k(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) \quad (13.57)$$

It has been proved that the matching kernel is not a Mercer kernel [5, 65].

Construction of the summation kernel or the matching kernel is computationally intensive. The number of base kernel computations is $T_m \times T_n$ for the summation kernel and $2 \times T_m \times T_n$ for the matching kernel. Hence, the computational complexity of both the summation kernel and the matching kernel is $\mathcal{O}(T^2)$, where T is the maximum of set cardinalities T_m and T_n . The summation kernel and the matching kernel give a measure of global similarity between a pair of examples.

13.5.5.3. Intermediate matching kernel

An intermediate matching kernel (IMK) [5] is constructed by matching the sets of local feature vectors using a set of virtual feature vectors. The set of virtual feature vectors is usually obtained using the training data of all the classes, as described later in this section. Let $\mathbf{V} = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_Q\}$ be the set of Q virtual feature vectors. For the q th virtual feature vector \mathbf{v}_q , the local feature vectors \mathbf{x}_{mq}^* and \mathbf{x}_{nq}^* in \mathbf{X}_m and \mathbf{X}_n respectively that are the closest to \mathbf{v}_q are determined as follows:

$$\mathbf{x}_{mq}^* = \arg \min_{\mathbf{x} \in \mathbf{X}_m} \mathcal{D}(\mathbf{x}, \mathbf{v}_q) \quad \text{and} \quad \mathbf{x}_{nq}^* = \arg \min_{\mathbf{x} \in \mathbf{X}_n} \mathcal{D}(\mathbf{x}, \mathbf{v}_q) \quad (13.58)$$

where $\mathcal{D}(\cdot, \cdot)$ is a function that measures the distance of a local feature vector in \mathbf{X}_m or \mathbf{X}_n to a virtual feature vector in \mathbf{V} . A pair of local feature vectors from \mathbf{X}_m and \mathbf{X}_n is selected for each of the virtual feature vectors in \mathbf{V} . A base kernel is computed for each of the Q pairs of selected local feature vectors. The IMK is computed as the sum of all the Q base kernel values as follows:

$$K_{\text{IMK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{q=1}^Q k(\mathbf{x}_{mq}^*, \mathbf{x}_{nq}^*) \quad (13.59)$$

The selection of the closest local feature vectors for a virtual feature vector involves computation of $T_m + T_n$ distance functions. Therefore, the

computation of IMK involves a total of $Q \times (T_m + T_n)$ computations of distance function \mathcal{D} , $Q \times (T_m + T_n)$ comparison operations to select the local feature vectors and Q computations of the base kernel. Hence, the computational complexity of IMK is $\mathcal{O}(QT)$, where T is the maximum of set cardinalities T_m and T_n . When Q is significantly smaller than T_m and T_n , the construction of IMK is computationally less intensive than constructing the summation kernel in (13.56). The IMK is a valid positive semidefinite kernel. This is because the base kernel is a valid positive semidefinite kernel and the sum of valid positive semidefinite kernels is a valid positive semidefinite kernel [49].

The choices for the set of virtual feature vectors leads to different versions of IMK as discussed in the following subsections.

Codebook-based IMK: In [5], the set of the centers of Q clusters formed from the training data of all classes is considered as the set of virtual feature vectors. The local feature vectors \mathbf{x}_{mq}^* and \mathbf{x}_{nq}^* in \mathbf{X}_m and \mathbf{X}_n that are the closest to \mathbf{v}_q , the center of the q th cluster, are determined as follows:

$$\mathbf{x}_{mq}^* = \arg \min_{\mathbf{x} \in \mathbf{X}_m} \|\mathbf{x} - \mathbf{v}_q\| \quad \text{and} \quad \mathbf{x}_{nq}^* = \arg \min_{\mathbf{x} \in \mathbf{X}_n} \|\mathbf{x} - \mathbf{v}_q\| \quad (13.60)$$

The Gaussian kernel $k(\mathbf{x}_{mq}^*, \mathbf{x}_{nq}^*) = \exp(-\delta \|\mathbf{x}_{mq}^* - \mathbf{x}_{nq}^*\|^2)$ is used as the base kernel. Here δ is the width parameter of the Gaussian kernel that is empirically chosen. As the centers of clusters are used to build a codebook, this IMK is called as the codebook-based IMK (CBIMK). Figure 13.11 illustrates the process of computing IMK between a pair of examples when the centers of clusters are considered as virtual feature vectors. The synthetic data of 3 classes is used for illustration. An example is represented by a set of 2-dimensional local feature vectors. The regions of 4 clusters and boundaries between the clusters formed for the training data of 3 classes using the k -means clustering method are shown in Figure 13.11(a). The process of computation of CBIMK for a pair of examples \mathbf{X}_m and \mathbf{X}_n belonging to the same class is illustrated in Figure 13.11(b). It is seen that the local feature vectors are mostly located around the centers of the clusters 1 and 2. It is noted that the local feature vectors selected from \mathbf{X}_m and \mathbf{X}_n using the center of each cluster are close to each other. The measure of closeness given by the value of base kernel is expected to be higher for local feature vectors selected by all the virtual feature vectors or cluster centers. This contributes to a higher value for CBIMK. The process of computation of CBIMK for a pair of examples \mathbf{X}_m and \mathbf{X}_l belonging to two different classes

is illustrated in Figure 13.11(c). It is seen that the local feature vectors of \mathbf{X}_m are located around the centers of the clusters 1 and 2, and the local feature vectors of \mathbf{X}_l are located around the centers of the clusters 3 and 4. The selected local feature vectors from \mathbf{X}_m and \mathbf{X}_l that are closest to the centers of clusters 1, 3 and 4 are located far from each other, as shown in Figure 13.11(c). The value of base kernel for these three pairs of local feature vectors is expected to be small. Therefore the value of CBIMK constructed in Figure 13.11(c) is expected to be lower compared to the value of CBIMK for the examples belonging to the same classes in Figure 13.11(b).

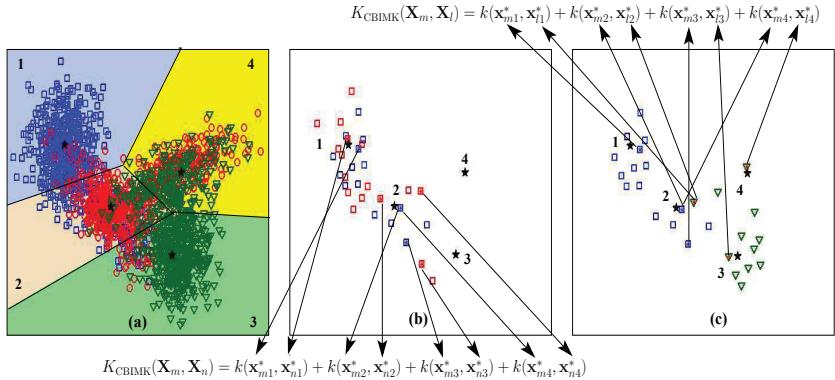


Fig. 13.11. Illustration of construction of CBIMK between a pair of sets of 2-dimensional local feature vectors, when the centers of clusters are considered as virtual feature vectors. (a) 2-dimensional local feature vectors belonging to the training data of three classes, denoted by \circ , \square , and \triangledown , are grouped into four clusters using the k -means clustering method. The regions of clusters and boundaries between clusters are also shown. The centers of the clusters considered as \mathbf{v}_q s are denoted using the symbol \star . (b) Illustration of computation of CBIMK between a pair of examples \mathbf{X}_m and \mathbf{X}_n belonging to the same class denoted by \square . The points corresponding to the local feature vectors of an example are shown in the same color. (c) Illustration of computation of CBIMK between a pair of examples \mathbf{X}_m and \mathbf{X}_l belonging to two different classes denoted by \square and \triangledown .

Construction of CBIMK uses only the information about the centers of clusters for representing the set of virtual feature vectors. A better representation for the set of virtual feature vectors can be provided by considering additional information about the clusters.

Gaussian mixture model based IMK: In [70] components of a class-independent Gaussian mixture model (CIGMM) built using the training data of all the classes is considered as a representation for the set of virtual feature vectors. This representation for the set of virtual feature vectors makes use of information in the mean vectors, covariance matrices and mixture coefficients of components of the CIGMM. The selection of a local feature vector using a component of CIGMM is based on the posterior probability of the component for that local feature vector. The posterior probability of q th component for a local feature vector \mathbf{x} , $P(q|\mathbf{x})$, called as the responsibility term $\gamma_q(\mathbf{x})$, is given by

$$P(q|\mathbf{x}) = \gamma_q(\mathbf{x}) = \frac{w_q \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)}{\sum_{q'=1}^Q w_{q'} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{q'}, \boldsymbol{\Sigma}_{q'})} \quad (13.61)$$

The local feature vectors \mathbf{x}_{mq}^* and \mathbf{x}_{nq}^* respectively in \mathbf{X}_m and \mathbf{X}_n are selected using the component q as

$$\mathbf{x}_{mq}^* = \arg \max_{\mathbf{x} \in \mathbf{X}_m} \gamma_q(\mathbf{x}) \quad \text{and} \quad \mathbf{x}_{nq}^* = \arg \max_{\mathbf{x} \in \mathbf{X}_n} \gamma_q(\mathbf{x}) \quad (13.62)$$

The CIGMM-based IMK, K_{CIGMMIMK} is computed as the sum of the values of the base kernels computed for the Q pairs of selected local feature vectors as in (13.59). Figure 13.12 illustrates the process of computing IMK between a pair of examples when the components of CIGMM are considered as the virtual feature vectors. The synthetic data of 3 classes is used for illustration. An example is represented by a set of 2-dimensional local feature vectors. The 4 components of CIGMM and their spreads are shown using the level curves in Figure 13.12(a). The process of computation of CIGMM-based IMK for a pair of examples \mathbf{X}_m and \mathbf{X}_n belonging to the same class is illustrated in Figure 13.12(b). Here, a local feature vector with the highest posterior probability corresponding to a component is selected for matching as opposed to a local feature vector closest to the center of a cluster in CBIMK. As the process of selecting local feature vectors is different, it is seen that the selected local feature vectors from \mathbf{X}_m and \mathbf{X}_n in Figure 13.12(b) are different from the local feature vectors selected in Figure 13.11(b). The local feature vectors selected from \mathbf{X}_m and \mathbf{X}_n corresponding to each component are located much closer to each other (especially the feature vectors selected using components 1 and 4) in Figure 13.12(b) than that of the local feature vectors selected in Figure 13.11(b). This contributes to a higher value for CIGMM-based IMK when the examples belong to the same classes. The process of computation of CIGMM-based IMK for a pair of examples \mathbf{X}_m and \mathbf{X}_l belonging to two

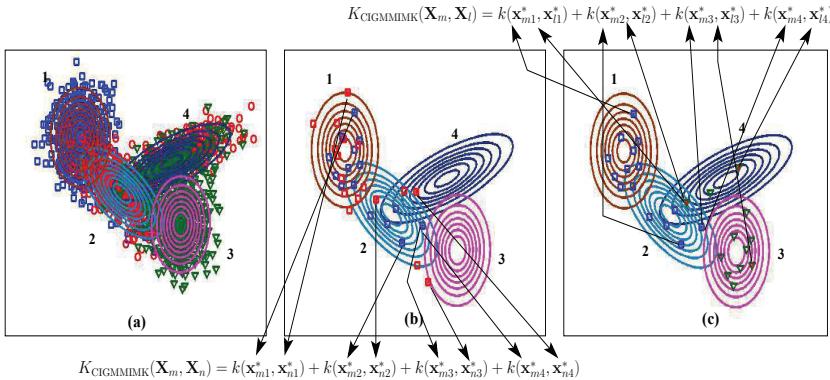


Fig. 13.12. Illustration of construction of CIGMM-based IMK between a pair of sets of 2-dimensional local feature vectors when the components of CIGMM are considered as virtual feature vectors. (a) CIGMM with four components built using the 2-dimensional local feature vectors belonging to the training data of three classes, denoted by \circ , \square , and ∇ . The level curves for different components of CIGMM are also shown. (b) Illustration of computation of CIGMM-based IMK between a pair of examples \mathbf{X}_m and \mathbf{X}_n belonging to the same class denoted by \square . The points corresponding to the local feature vectors of an example are shown in the same color. (c) Illustration of computation of CIGMM-based IMK between a pair of examples \mathbf{X}_m and \mathbf{X}_l belonging to two different classes denoted by \square and ∇ .

different classes is illustrated in Figure 13.12(c). It is seen that the selected local feature vectors from \mathbf{X}_m and \mathbf{X}_l corresponding to each component in Figure 13.12(c) are different from the local feature vectors selected in Figure 13.11(c) and hence the value of CIGMM-based IMK is different from that of the CBIMK for the examples belonging to the two different classes.

13.6. Dynamic kernels for sequences of local feature vectors

In this section, we present the score space kernels, probability product kernel, alignment kernels, intermediate matching kernel and pyramid matching kernel for sequences of local feature vectors.

13.6.1. Score space kernels

Score space kernels such as Fisher kernel (FK) [3] and likelihood ratio kernel (LRK) [63] for sequential patterns use a HMM for mapping a sequence onto a Fisher score space.

13.6.1.1. Fisher kernel for sequences of local feature vectors

Fisher kernel [3] for sequential patterns uses an explicit expansion into a kernel feature space defined by the HMM-based likelihood score space where a sequence of local feature vectors is represented as a fixed dimensional Fisher score vector. Likelihood score space for a class is obtained by the first order derivatives of the log likelihood of the HMM for that class with respect to the HMM parameters such as the state transition probabilities and the state-specific GMM parameters. For a sequence of local feature vectors $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T)$, the first order derivatives of the log-likelihood for a class c , $\ln p(\mathbf{X}|\lambda_c)$, are given by the gradient vector of the log-likelihood for class c with respect to parameters of HMM λ_c as follows:

$$\Phi_c(\mathbf{X}) = \nabla \ln p(\mathbf{X}|\lambda_c) \quad (13.63)$$

For a sequence of local feature vectors \mathbf{X} , the gradient vector with respect to the mean vector of the q th component of the GMM of the i th state, $\boldsymbol{\mu}_{iq,c}$ is given by [63]

$$\varphi_{iq,c}^{(\boldsymbol{\mu})}(\mathbf{X}) = \sum_{t=1}^T \gamma_{iq,c}(\mathbf{x}_t) \mathbf{z}_{iq,c}^\top \quad (13.64)$$

Here $\mathbf{z}_{iq,c} = (\mathbf{x}_t - \boldsymbol{\mu}_{iq,c})^\top \Sigma_{iq,c}^{-1}$ and $\gamma_{iq,c}(\mathbf{x}_t)$ is the responsibility term corresponding to the probability of being in state i at time t and generating \mathbf{x}_t using the component q given by

$$\gamma_{iq,c}(\mathbf{x}_t) = \frac{w_{iq,c} \mathcal{N}(\mathbf{x}_t | \boldsymbol{\mu}_{iq,c}, \boldsymbol{\Sigma}_{iq,c})}{\sum_{j=1}^Q w_{ij,c} \mathcal{N}(\mathbf{x}_t | \boldsymbol{\mu}_{ij,c}, \boldsymbol{\Sigma}_{ij,c})} \quad (13.65)$$

Here $w_{iq,c}$ is the mixture coefficient and $\mathcal{N}(\mathbf{x} | \boldsymbol{\mu}_{iq,c}, \boldsymbol{\Sigma}_{iq,c})$ is the normal density with mean $\boldsymbol{\mu}_{iq,c}$ and covariance $\boldsymbol{\Sigma}_{iq,c}$. The gradient vector of the log-likelihood with respect to $\boldsymbol{\Sigma}_{iq,c}$ is given by

$$\varphi_{iq,c}^{(\boldsymbol{\Sigma})}(\mathbf{X}) = \sum_{t=1}^T \gamma_{iq,c}(\mathbf{x}_t) \frac{1}{2} \left[-\text{vec}(\boldsymbol{\Sigma}_{iq,c}^{-1})^\top + \{\mathbf{z}_{iq,c} \otimes \mathbf{z}_{iq,c}\} \right]^\top \quad (13.66)$$

where $\text{vec}(\boldsymbol{\Sigma}_{iq,c}) = [\sigma_{11,c}^{(iq)}, \sigma_{12,c}^{(iq)}, \dots, \sigma_{dd,c}^{(iq)}]^\top$ and $\{\mathbf{z}_{iq,c} \otimes \mathbf{z}_{iq,c}\} = [z_{1iq,c} \mathbf{z}_{iq,c}^\top, z_{2iq,c} \mathbf{z}_{iq,c}^\top, \dots, z_{diq,c} \mathbf{z}_{iq,c}^\top]^\top$. Here d is the dimension of the feature vector. The gradient value of the log-likelihood for a class c with respect to $w_{iq,c}$ is given by [63]

$$\varphi_{iq,c}^{(w)}(\mathbf{X}) = \sum_{t=1}^T \gamma_{iq,c}(\mathbf{x}_t) \left[\frac{1}{w_{iq,c}} - \frac{\gamma_{i1,c}(\mathbf{x}_t)}{w_{i1,c} \gamma_{iq,c}(\mathbf{x}_t)} \right] \quad (13.67)$$

The Fisher score vector with respect to the parameters of the q th component of the GMM for state i of a class c is obtained as a supervector of gradient vectors of the log-likelihood for that component and is given by

$$\tilde{\Phi}_{iq,c}(\mathbf{X}) = \left[\varphi_{iq,c}^{(\mu)}(\mathbf{X})^\top, \varphi_{iq,c}^{(\Sigma)}(\mathbf{X})^\top, \varphi_{iq,c}^{(w)}(\mathbf{X})^\top \right]^\top \quad (13.68)$$

The Fisher score vector with respect to the parameters of the i th state of the HMM for class c is a supervector of all the Fisher score vectors obtained from the Q_i components and is given by

$$\hat{\Phi}_{i,c}(\mathbf{X}) = \left[\tilde{\Phi}_{i1,c}(\mathbf{X})^\top, \tilde{\Phi}_{i2,c}(\mathbf{X})^\top, \dots, \tilde{\Phi}_{iQ_i,c}(\mathbf{X})^\top \right]^\top \quad (13.69)$$

Now the sequence of local feature vectors \mathbf{X} is represented as a fixed dimensional supervector $\Phi_c(\mathbf{X})$ of all the Fisher score vectors obtained from the N states of the HMM for class c as follows:

$$\Phi_c(\mathbf{X}) = \left[\hat{\Phi}_{1,c}(\mathbf{X})^\top, \hat{\Phi}_{2,c}(\mathbf{X})^\top, \dots, \hat{\Phi}_{N,c}(\mathbf{X})^\top \right]^\top \quad (13.70)$$

The dimension of Fisher score vector is $D=N \times Q_i \times (d + d^2 + 1)$. The Fisher kernel for class c between two sequences of local feature vectors \mathbf{X}_m and \mathbf{X}_n is computed as

$$K_{FK}(\mathbf{X}_m, \mathbf{X}_n) = \Phi_c(\mathbf{X}_m)^\top \mathbf{F}_c^{-1} \Phi_c(\mathbf{X}_n) \quad (13.71)$$

Here \mathbf{F}_c is the Fisher information matrix for class c given by

$$\mathbf{F}_c = \frac{1}{L_c} \sum_{l=1}^{L_c} \Phi_c(\mathbf{X}_l) \Phi_c(\mathbf{X}_l)^\top \quad (13.72)$$

where L_c is the number of training examples for class c . Fisher kernel is a class-specific kernel, as it uses the HMM for a class to compute the kernel for that class. Hence the SVM classifier with Fisher kernel uses the one-against-the-rest approach to multi-class pattern classification.

The computation of each of the three gradient vectors involves $N \times Q_i \times (T_m + T_n)$ operations. The computation of Fisher information matrix involves $L \times D^2 + L$ operations and kernel computation involves $D^2 + D$ operations. Hence, the computational complexity of Fisher kernel is $\mathcal{O}(NQT + LD^2 + L + D^2 + D)$. Specifically, the computation of Fisher information matrix and its inverse is computationally intensive. For example, in a 12-state HMM with 2 mixtures per state and for the 39-dimensional feature vectors, the number of derivatives to be computed is approximately 37500, and the dimension of Fisher information matrix is 37500x37500.

13.6.1.2. Likelihood ratio kernel

Likelihood ratio kernel [63] for sequences of local feature vectors uses an explicit expansion into a kernel feature space defined by the HMM-based likelihood ratio score space where a set of local feature vectors is represented as a fixed dimensional Fisher score vector. Likelihood ratio score space is obtained by the first order derivatives of the ratio of the log likelihoods of a pair of classes with respect to the HMM parameters. Let c_1 and c_2 be the two classes. For a sequence of local feature vectors $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T)$, the ratio of the log likelihoods of the two classes is given by $p(\mathbf{X}|\lambda_{c_1})/p(\mathbf{X}|\lambda_{c_2})$. For \mathbf{X} , the first order derivative of the ratio of log likelihoods for a pair of classes c_1 and c_2 with respect to parameters of HMMs λ_{c1} and λ_{c2} is given by

$$\varphi_{iq,c}(\mathbf{X}) = \nabla \ln \frac{p(\mathbf{X}|\lambda_{c_1})}{p(\mathbf{X}|\lambda_{c_2})} \quad (13.73)$$

Rest of the procedure for computing the likelihood ratio kernel is the same as that for the Fisher kernel. The only difference is that, in Fisher kernel the kernel feature space is the likelihood score space and a kernel is computed for every class. In the likelihood ratio kernel, the kernel feature space is the likelihood ratio score space, and a kernel is computed for every pair of classes. The computational complexity of the likelihood ratio kernel is almost the same as that of Fisher kernel.

13.6.2. Probability product kernel

Another kernel designed for sequential patterns is the probability product kernel [64] that matches the distributions derived from the sequences of local feature vectors. Here, each sequence of local feature vectors is represented by an HMM. The Kullback-Leibler (KL) divergence [71] is used as a measure of dissimilarity between two HMMs. The probability product kernel is obtained by kernelizing the KL divergence between two HMMs. However, in speech recognition tasks such as E-set recognition and CV segment recognition, patterns are extracted from short duration (200 to 500 milliseconds) segments of speech that can not be mapped onto distributions, i.e., building an HMM for each example is difficult.

13.6.3. Alignment kernels

The alignment kernels such as dynamic time warping kernel (DTWK) [67], dynamic time alignment kernel (DTAK) [60], global alignment kernel

(GAK) [68] and triangular global alignment kernel (TGAK) [61] compute a kernel between a pair of sequences of local feature vectors using the dynamic time warping (DTW) method.

13.6.3.1. Dynamic time warping kernel

A well known distance measure used for comparing two sequential patterns is the dynamic time warping (DTW) distance [8]. Let $\mathbf{X}_m = (\mathbf{x}_{m1}, \mathbf{x}_{m2}, \dots, \mathbf{x}_{mT_m})$ and $\mathbf{X}_n = (\mathbf{x}_{n1}, \mathbf{x}_{n2}, \dots, \mathbf{x}_{nT_n})$ be the sequences of local feature vectors for two examples. Computation of the DTW distance involves aligning two sequences of different lengths. An alignment between two sequences \mathbf{X}_m and \mathbf{X}_n is represented by a pair of vectors (ω_1, ω_2) of length $T_a \leq T_m + T_n - 1$ such that $1 = \omega_1(1) \leq \dots \leq \omega_1(T_a) = T_m$ and $1 = \omega_2(1) \leq \dots \leq \omega_2(T_a) = T_n$. The DTW distance is computed as the distance along an optimal path that minimizes the accumulated measure of dissimilarity as,

$$\text{DTW}(\mathbf{X}_m, \mathbf{X}_n) = \min_{\omega_1, \omega_2} \sum_{p=1}^{T_a} \|\mathbf{x}_{m\omega_1(p)} - \mathbf{x}_{n\omega_2(p)}\|^2 \quad (13.74)$$

The optimization problem in (13.74) can be solved efficiently using dynamic programming. Let the recursive function $G(\mathbf{x}_{mt}, \mathbf{x}_{nt'})$ be defined as follows:

$$G(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) = \min \begin{cases} G(\mathbf{x}_{mt-1}, \mathbf{x}_{nt'}) + \text{dist}(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) \\ G(\mathbf{x}_{mt-1}, \mathbf{x}_{nt'-1}) + \text{dist}(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) \\ G(\mathbf{x}_{mt}, \mathbf{x}_{nt'-1}) + \text{dist}(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) \end{cases} \quad (13.75)$$

The bounding values of the recursive function are considered as $G(\mathbf{x}_{m0}, \mathbf{x}_{n0}) = 1$ and $G(\mathbf{x}_{m0}, \mathbf{x}_{nt'}) = G(\mathbf{x}_{mt}, \mathbf{x}_{n0}) = 0$. The $\text{dist}(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) = \|\mathbf{x}_{mt} - \mathbf{x}_{nt'}\|^2$ is the Euclidean distance function indicating the measure of dissimilarity between the local aligned vectors. The DTW distance between the two sequences \mathbf{X}_m and \mathbf{X}_n is given by

$$\text{DTW}(\mathbf{X}_m, \mathbf{X}_n) = \frac{1}{T_m + T_n} G(\mathbf{x}_{mT_m}, \mathbf{x}_{nT_n}) \quad (13.76)$$

In [67], the DTW kernel between \mathbf{X}_m and \mathbf{X}_n is defined as

$$K_{\text{DTWK}}(\mathbf{X}_m, \mathbf{X}_n) = \exp(-\delta \text{DTW}(\mathbf{X}_m, \mathbf{X}_n)) \quad (13.77)$$

Here δ is the user defined parameter that is empirically chosen. The DTW kernel is shown to be positive definite kernel only under some favorable conditions [61, 68]. The computational complexity of the DTW kernel between \mathbf{X}_m and \mathbf{X}_n is $\mathcal{O}(T^2)$, where T is the maximum of T_m and T_n .

13.6.3.2. Dynamic time alignment kernel

The dynamic time alignment kernel (DTAK) [60] between two sequences is computed by computing the DTW distance between them in a kernel feature space. Let $\Phi(\mathbf{X}_m) = (\phi(\mathbf{x}_{m1}), \phi(\mathbf{x}_{m2}), \dots, \phi(\mathbf{x}_{mT_m}))$ and $\Phi(\mathbf{X}_n) = (\phi(\mathbf{x}_{n1}), \phi(\mathbf{x}_{n2}), \dots, \phi(\mathbf{x}_{nT_n}))$ be the representation of sequences \mathbf{X}_m and \mathbf{X}_n in the kernel feature space of a basic kernel such as Gaussian kernel. The DTW distance in the kernel feature space is computed as $\text{DTW}(\Phi(\mathbf{X}_m), \Phi(\mathbf{X}_n))$ using (13.76). The dynamic time alignment kernel is computed as

$$K_{\text{DTAK}}(\mathbf{X}_m, \mathbf{X}_n) = \exp(-\delta \text{DTW}(\Phi(\mathbf{X}_m), \Phi(\mathbf{X}_n))) \quad (13.78)$$

The dynamic time alignment kernel is shown to be positive definite kernel only under some favorable conditions [61, 68]. The computational complexity of the dynamic time alignment kernel between \mathbf{X}_m and \mathbf{X}_n is $\mathcal{O}(T^2)$, where T is the maximum of the T_m and T_n .

13.6.3.3. Global alignment kernel

The DTW distance uses an optimal alignment path whose cost is the minimum of the costs of all the alignments. The global alignment kernel (GAK) [68] uses the soft-minimum of the costs of all the alignments to define a positive definite kernel. Let $\mathbf{X}_m = (\mathbf{x}_{m1}, \mathbf{x}_{m2}, \dots, \mathbf{x}_{mT_m})$ and $\mathbf{X}_n = (\mathbf{x}_{n1}, \mathbf{x}_{n2}, \dots, \mathbf{x}_{nT_n})$ be the sequences of local feature vectors for two examples. The recursive function $G_a(\mathbf{x}_{mt}, \mathbf{x}_{nt'})$ used in computation of GAK is defined as follows:

$$\begin{aligned} G_a(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) &= k(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) \left[G_a(\mathbf{x}_{mt}, \mathbf{x}_{nt'-1}) \right. \\ &\quad \left. + G_a(\mathbf{x}_{mt-1}, \mathbf{x}_{nt'}) + G_a(\mathbf{x}_{mt-1}, \mathbf{x}_{nt'-1}) \right] \end{aligned} \quad (13.79)$$

Here, $k(\mathbf{x}_{mt}, \mathbf{x}_{nt'})$ is the Gaussian kernel function indicating the similarity between the aligned local feature vectors \mathbf{x}_{mt} and $\mathbf{x}_{nt'}$. The bounding values of the recursive function are considered as $G_a(\mathbf{x}_{m0}, \mathbf{x}_{n0}) = 1$ and $G_a(\mathbf{x}_{m0}, \mathbf{x}_{nt'}) = G_a(\mathbf{x}_{mt}, \mathbf{x}_{n0}) = 0$. The GAK between \mathbf{X}_m and \mathbf{X}_n is given as follows:

$$K_{\text{GAK}}(\mathbf{X}_m, \mathbf{X}_n) = G_a(\mathbf{x}_{mT_m}, \mathbf{x}_{nT_n}) \quad (13.80)$$

Since the GAK considers all possible alignments between two sequences, it is computationally intensive. The computational complexity of GAK between \mathbf{X}_m and \mathbf{X}_n is $\mathcal{O}(T^2)$, where T is the maximum of T_m and T_n .

13.6.3.4. Triangular global alignment kernel

The triangular global alignment kernel (TGAK) [61] is an extension to GAK to speed up its computation. The computation of TGAK uses a triangular kernel [72], computed on the position of \mathbf{x}_{mt} and $\mathbf{x}_{nt'}$ in their respective sequences, as a weight for the basic kernel $k(\mathbf{x}_{mt}, \mathbf{x}_{nt'})$ in (13.79). In [61], the triangular kernel, $\zeta(t, t')$, is defined as

$$\zeta(t, t') = \begin{cases} 1 - \frac{|t-t'|}{r} & \text{if } |t-t'| \leq r \\ 0 & \text{otherwise} \end{cases} \quad (13.81)$$

where $1 < r < \frac{T_m+T_n}{2}$ is the user defined parameter that is empirically chosen. Now the basic kernel between \mathbf{x}_{mt} and $\mathbf{x}_{nt'}$ in (13.79) is modified as

$$\hat{k}(\mathbf{x}_{mt}, \mathbf{x}_{nt'}) = \frac{\zeta(t, t')k(\mathbf{x}_{mt}, \mathbf{x}_{nt'})}{2 - \zeta(t, t')k(\mathbf{x}_{mt}, \mathbf{x}_{nt'})} \quad (13.82)$$

This ensures that the value of basic kernel in (13.79) is zero when $|t-t'| > r$ and hence reduces the number of computations to obtain GAK. The GAK then is computed using the modified basic kernel to obtain TGAK, $K_{\text{TGAK}}(\mathbf{X}_m, \mathbf{X}_n)$ between \mathbf{X}_m and \mathbf{X}_n , as in (13.80). It is shown in [61] that the computational complexity of TGAK between \mathbf{X}_m and \mathbf{X}_n is less than $\mathcal{O}(T^2)$, where T is the maximum of T_m and T_n .

13.6.4. HMM-based intermediate matching kernel for sequences of continuous valued feature vectors

In Section 13.5.5.3 an intermediate matching kernel (IMK) is constructed by matching the examples represented as sets of local feature vectors using a set of virtual feature vectors. Here, components of a class-independent Gaussian mixture model (CIGMM) are used as the set of virtual feature vectors to select the local feature vectors and then compute base kernels. The GMM-based IMK for a pair of examples is computed as a sum of base kernels. A base kernel is computed on a pair of local feature vectors selected by matching all the local feature vectors of the two examples with a component of the GMM. A local feature vector from an example with the maximum value of responsibility term for a component is selected. The GMM-based IMK is suitable for matching the examples represented as sets of feature vectors. As it does not use the sequence information in matching the examples, the GMM-based IMK is not suitable for sequential pattern classification as in acoustic modeling of subword units of speech.

In [73] the hidden Markov model (HMM) based IMK (HIMK) is proposed for matching the examples represented as sequences of feature vectors. In speech recognition tasks, a continuous-density HMM (CDHMM) is built for each class (a subword unit) using the sequences of feature vectors extracted from the training examples (utterances) of the class. The CDHMM for a subword unit is an N -state, left-to-right HMM with a state-specific GMM used to model the observation probability distribution of each state. The parameters of a CDHMM include the initial state probabilities, state transition probabilities, and the parameters of state-specific GMMs. The Baum-Welch method is the maximum likelihood method used for estimation of the parameters of a CDHMM. Two approaches to design the CDHMM based intermediate matching kernel for matching two sequences of feature vectors are proposed in [73]. In these approaches, the class-specific left-to-right CDHMM is used to construct an HIMK for the SVM of that class.

13.6.4.1. HMM-based IMK using subsequence matching

In this approach, the HIMK for two sequences of feature vectors is constructed by first segmenting each sequence into N subsequences associated with N states of an HMM, and then computing a state-specific GMM-based IMK for the subsequences associated with each state. Let $\mathbf{X}_m = (\mathbf{x}_{m1}, \mathbf{x}_{m2}, \dots, \mathbf{x}_{mT_m})$ and $\mathbf{X}_n = (\mathbf{x}_{n1}, \mathbf{x}_{n2}, \dots, \mathbf{x}_{nT_n})$ be the sequences of feature vectors for two examples. Let $\mathbf{S}_m = (s_1^m, s_2^m, \dots, s_{T_m}^m)$ and $\mathbf{S}_n = (s_1^n, s_2^n, \dots, s_{T_n}^n)$ be the optimal sequences of states of HMM obtained using the Viterbi algorithm for \mathbf{X}_m and \mathbf{X}_n respectively. For a left-to-right HMM, the states in the optimal state sequences are non-decreasing in order. Therefore, the states at time t and $t+1$ for \mathbf{X}_m and \mathbf{X}_n satisfy the property that $s_t^m \leq s_{t+1}^m$ and $s_t^n \leq s_{t+1}^n$ respectively. This property of optimal state sequences is used to segment each of the sequences of feature vectors into N subsequences of feature vectors. Let \mathbf{X}_m^i , $i = 1, 2, \dots, N$, be the N subsequences of \mathbf{X}_m . The feature vectors in the subsequence \mathbf{X}_m^i are aligned with the state i . In acoustic modeling of a speech unit using an HMM, each state is associated with an acoustic event. Therefore all the feature vectors in the subsequence \mathbf{X}_m^i are associated with the acoustic event of state i . The sequence information among the feature vectors within a subsequence is not important for matching. Therefore, the state-specific GMM-based IMK is used for matching the subsequences of two examples aligned with each state.

Let Ψ_i be the GMM of state i in the HMM. Let Q_i be the number of

components in Ψ_i . The posterior probability of q th component of GMM Ψ_i for a local feature vector \mathbf{x} is given as the responsibility term $\gamma_{iq}(\mathbf{x})$, as follows:

$$\gamma_{iq}(\mathbf{x}) = \frac{w_{iq}\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{iq}, \boldsymbol{\Sigma}_{iq})}{\sum_{q'=1}^{Q_i} w_{iq'}\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{iq'}, \boldsymbol{\Sigma}_{iq'})} \quad (13.83)$$

where w_{iq} is the mixture coefficient, $\boldsymbol{\mu}_{iq}$ is the mean and $\boldsymbol{\Sigma}_{iq}$ is the covariance matrix for the q th component of Ψ_i . The local feature vectors \mathbf{x}_{miq}^* and \mathbf{x}_{niq}^* respectively in the subsequences \mathbf{X}_m^i and \mathbf{X}_n^i are selected using the component q of Ψ_i as follows:

$$\mathbf{x}_{miq}^* = \arg \max_{\mathbf{x} \in \mathbf{X}_m^i} \gamma_{iq}(\mathbf{x}) \quad \text{and} \quad \mathbf{x}_{niq}^* = \arg \max_{\mathbf{x}' \in \mathbf{X}_n^i} \gamma_{iq}(\mathbf{x}') \quad (13.84)$$

The state-specific GMM-based IMK for the subsequences \mathbf{X}_m^i and \mathbf{X}_n^i , $K_{\Psi_i}(\mathbf{X}_m^i, \mathbf{X}_n^i)$, is computed as the sum of the base kernels computed for the Q_i pairs of selected feature vectors as

$$K_{\Psi_i}(\mathbf{X}_m^i, \mathbf{X}_n^i) = \sum_{q=1}^{Q_i} k(\mathbf{x}_{miq}^*, \mathbf{x}_{niq}^*) \quad (13.85)$$

The state-specific GMM-based IMKs are computed for subsequences associated with the N states. The subsequence matching based HIMK (SSHIMK) for sequences \mathbf{X}_m and \mathbf{X}_n is computed as a combination of the state-specific GMM-based IMKs as follows:

$$K_{\text{SSHIMK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{i=1}^N K_{\Psi_i}(\mathbf{X}_m^i, \mathbf{X}_n^i) \quad (13.86)$$

Figure 13.13 illustrates the process of computing the SSHIMK between a pair of sequences of feature vectors for $N = 3$ and $Q_i = 3$, $i = 1, 2, 3$.

The construction of a CDHMM used in computation of SSHIMK is a one time process. The computational complexity for segmenting the sequences into N subsequences associated with N states of HMM using the Viterbi algorithm for a pair of sequences \mathbf{X}_m and \mathbf{X}_n is $\mathcal{O}(N^2T)$, where T is the maximum of the lengths of sequences T_m and T_n . Then, the computation of GMM-based IMK for the state i involves a total of $Q_i \times (T_m^i + T_n^i)$ computations of the posterior probability of components of i th state GMM for each local feature vector in the two subsequences aligned to that state, $Q_i \times (T_m^i + T_n^i)$ comparison operations to select the local feature vectors and Q_i computations of the base kernel. Here, T_m^i and T_n^i are the lengths of the subsequences \mathbf{X}_m^i and \mathbf{X}_n^i aligned to state i . Hence the computational complexity of SSHIMK is $\mathcal{O}(NQT')$ plus the computational

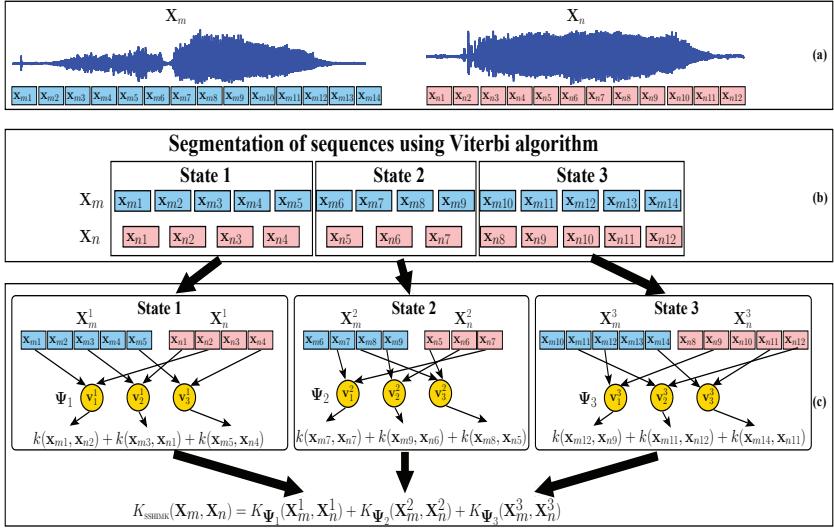


Fig. 13.13. Illustration of construction of SSHIMK between a pair of sequences of feature vectors \mathbf{X}_m and \mathbf{X}_n . A 3-state left-to-right HMM with a 3-component GMM for each state is considered in the illustration. (a) Sequence of feature vectors representation of two speech signals. (b) Segmentation of each sequence of feature vectors into 3 subsequences of feature vectors using Viterbi alignment. (c) Computation of HIMK between a pair of examples. The Ψ_i denotes the GMM for the state i and v_q^i denotes the q th component of Ψ_i .

complexity for Viterbi alignment. Here T' is the maximum of the lengths of the subsequences aligned to the N states and Q is the maximum of the number of components in state-specific GMMs.

The Gaussian kernel which is a valid positive semidefinite kernel is used as the base kernel. The state-specific GMM-based IMK is a valid positive semidefinite kernel because the sum of valid positive semidefinite kernels (base kernels) is a valid positive semidefinite kernel [49]. The SSHIMK is also a valid positive semidefinite kernel because the sum of valid positive semidefinite kernels (state-specific GMM-based IMKs) is a valid positive semidefinite kernel [49]. Next, we present another approach to construct an IMK that does not involve segmentation of a sequence into subsequences.

13.6.4.2. HMM-based IMK using complete sequence matching

In this approach, an IMK for a pair of sequences of feature vectors is constructed by matching a pair of complete sequences and as a combination

of base kernels. A base kernel corresponding to every component of the GMM of each state is computed between a pair of local feature vectors, one from each sequence, selected with the highest value of responsibility term for that component. Let $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t, \dots, \mathbf{x}_T)$ be a sequence of feature vectors. For a continuous density HMM λ , the probability of being in state i at time t and the q th component of GMM Ψ_i of state i accounting for the local feature vector \mathbf{x}_t in the sequence \mathbf{X} is given by

$$R_{iq}(\mathbf{x}_t | \mathbf{X}, \lambda) = \nu_{it} \gamma_{iq}(\mathbf{x}_t) \quad (13.87)$$

Here the term ν_{it} is the probability of being in state i at time t given \mathbf{X} and λ , i.e.,

$$\nu_{it} = P[s_t = i | \mathbf{X}, \lambda] \quad (13.88)$$

The term ν_{it} can be computed using the parameters of HMM λ [8]. The term $\gamma_{iq}(\mathbf{x}_t)$ in (13.87) is the responsibility of the component q of GMM Ψ_i for a feature vector \mathbf{x}_t as in (13.83). For the sequences \mathbf{X}_m and \mathbf{X}_n , the local feature vectors \mathbf{x}_{miq}^* and \mathbf{x}_{niq}^* to be matched are selected as follows:

$$\mathbf{x}_{miq}^* = \arg \max_{\mathbf{x}_t \in \mathbf{X}_m} R_{iq}(\mathbf{x}_t | \mathbf{X}_m, \lambda) \quad \text{and} \quad \mathbf{x}_{niq}^* = \arg \max_{\mathbf{x}_{t'} \in \mathbf{X}_n} R_{iq}(\mathbf{x}_{t'} | \mathbf{X}_n, \lambda) \quad (13.89)$$

The complete sequence matching based HIMK (CSHIMK) for \mathbf{X}_m and \mathbf{X}_n is computed as

$$K_{\text{CSHIMK}}(\mathbf{X}_m, \mathbf{X}_n) = \sum_{i=1}^N \sum_{q=1}^{Q_i} k(\mathbf{x}_{miq}^*, \mathbf{x}_{niq}^*) \quad (13.90)$$

where Q_i is the number of components in Ψ_i . Figure 13.14 illustrates the process of computing the CSHIMK between a pair of examples using a HMM λ with $N = 3$ and $Q_i = 3$, $i = 1, 2, 3$.

It should be noted that there is no temporal ordering among the local feature vectors selected using the components of GMM for a state. However, temporal order is preserved across the states. In acoustic modeling of a speech unit using HMM, each state is associated with an acoustic event. Therefore all the feature vectors selected using the components of a state-specific GMM are associated with the acoustic event of that state. For example, in Figure 13.14 the local feature vectors \mathbf{x}_{n8} , \mathbf{x}_{n12} and \mathbf{x}_{n11} from the sequence \mathbf{X}_n selected using the components 1, 2 and 3 of GMM for state 3 do not follow the temporal order. It is also seen that, from the sequence \mathbf{X}_n , the local feature vectors \mathbf{x}_{n1} , \mathbf{x}_{n2} and \mathbf{x}_{n3} are selected using Ψ_1 of state 1, \mathbf{x}_{n6} , \mathbf{x}_{n7} and \mathbf{x}_{n5} are selected using Ψ_2 of state 2 and \mathbf{x}_{n8} , \mathbf{x}_{n12}

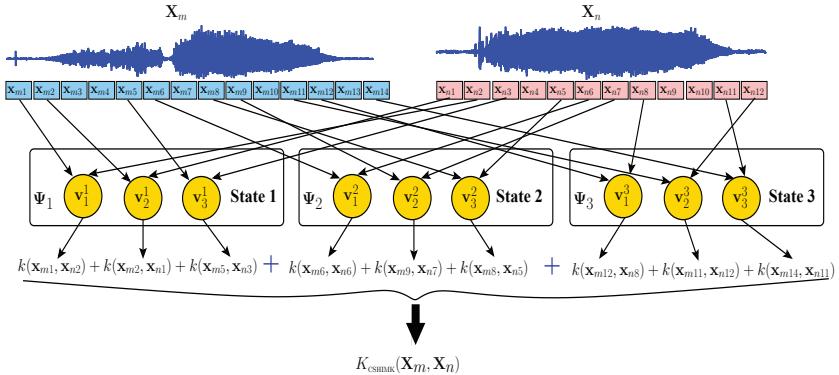


Fig. 13.14. Illustration of construction of CSHIMK between a pair of sequences of feature vectors \mathbf{X}_m and \mathbf{X}_n . A 3-state left-to-right HMM with a 3-component GMM for each state is considered in the illustration. The Ψ_i denotes the GMM for state i and v_q^i denotes the q th component of Ψ_i . The feature vectors for matching are selected using the values of $R_{iq}(\mathbf{x}_t | \mathbf{X}, \lambda)$.

and \mathbf{x}_{n11} are selected using Ψ_3 of state 3. This shows that the temporal order is preserved across the states. This behavior of the complete sequence matching based HIMK is due to the left-to-right architecture of the HMM used to build the kernel. For a left-to-right HMM, the alignment of local feature vectors with the states will be in the monotonical (non-decreasing) order of states. It may be noted that there is no monotonic order in the alignment of local feature vectors with the components of the GMM of a state.

The construction of a CDHMM used in computation of the CSHIMK is a one time process. The computation of CSHIMK involves a total of $\sum_{i=1}^N Q_i \times (T_m + T_n)$ computations for computing the posterior probability of each component in each of the state-specific GMMs for the local feature vectors in the two sequences, $\sum_{i=1}^N Q_i \times (T_m + T_n)$ comparison operations to select the local feature vectors and NQ_i computations of the base kernel. Hence, the computational complexity of CSHIMK is $\mathcal{O}(NQT)$, where T is the maximum of the lengths of sequences T_m and T_n , and Q is the maximum of the number of components in state-specific GMMs.

The Gaussian kernel is considered as the base kernel which is a valid positive semidefinite kernel. The CSHIMK is a valid positive semidefinite kernel, because the sum of valid positive semidefinite kernels is a valid positive semidefinite kernel [49].

The main differences between the subsequence matching based approach and the complete sequence matching based approach for construction of HIMK are as follows: The first method uses the Viterbi alignment to find the optimal state sequence for a given observation sequence, and does the hard assignment of each local feature vector in the observation sequence to one of the states as per the alignment. Accordingly, the observation sequence is segmented into subsequences with each subsequence being associated with a state. The second method does not involve finding the optimal state sequence and segmentation of observation sequence into subsequences. The second method chooses a local feature vector for which the value of R_{iq} (as defined in (13.87)) is maximum, and associates that feature vector with the component q of the GMM of state i . Therefore, the second method also does the hard assignment to a component of GMM of a state, but not to a state. Also, the probability of being in state i at time t is computed using all possible state sequences, and not using only the optimal state sequence. It is observed that the values of the HIMK obtained using the two methods for a pair of examples are not the same, indicating that the two methods are not actually the same.

13.7. Studies on classification and matching of varying length patterns represented as sets of local feature vectors

In this Section, we present our studies on speech emotion recognition, speaker identification, scene image classification, scene image annotation and retrieval using varying length patterns represented as sets of local feature vectors.

13.7.1. *Speech emotion recognition and speaker identification using dynamic kernel based SVMs*

Speech emotion recognition task involves automatically identifying the emotional state of a speaker from his/her voice. Speaker identification task involves identifying a speaker among a known set of speakers using a speech utterance produced by the speaker. We present our studies on speech emotion recognition and speaker identification using the SVM-based systems with different dynamic kernels for sets of local feature vectors. We compare their performance with that of the GMM-based systems.

13.7.1.1. Databases and features used in the studies

The Berlin emotional speech database [74] and the German FAU Aibo emotion corpus (FAU-AEC) [75] are used for studies on speech emotion recognition task. The NIST speaker recognition (SRE) corpora [76, 77] are used for studies on speaker identification task.

Databases for emotion recognition from speech: The Berlin emotional speech database (Emo-DB) [74] consists of emotional utterances spoken by ten professional native German actors (Five female and five male) for ten sentences in German language. The speech was recorded with 16-bit precision and at a sampling rate of 22kHz. The duration of the utterances varies from two to eight seconds. This database contains 494 utterances belonging to the following seven emotional categories with the number of utterances for the category given in parentheses: fear (55), disgust (38), happiness (64), boredom (79), neutral (78), sadness (53), and anger (127). The multi-speaker speech emotion recognition accuracy presented is the average classification accuracy along with 95% confidence interval obtained for 5-fold stratified cross-validation.

The German FAU-AEC [75] includes 8.9 hours of spontaneous, emotionally colored children's speech. It comprises recordings of 51 German children at the age of 10 to 13 years from two different schools. Among 51 children, 30 are female and 21 are male. The speech was recorded with 16-bit precision and at a sampling rate of 48kHz. The data is then down sampled to 16kHz. The corpus includes 13,642 audio files with eleven emotions. For our studies, four super classes of emotions such as anger, emphatic, neutral, and motherese are considered. If the whole corpus is used for classification, these four super classes are highly unbalanced. Hence, we have considered an almost balanced subset of the corpus defined for these four classes by CEICES of the Network of Excellence HUMAINE funded by European Union [75]. This subset includes a total of 4543 speech utterances. We perform the classification at the chunk (speech utterance) level in the Aibo chunk set which contains all those chunks that contain at least one word. The speaker-independent speech emotion recognition accuracy presented is the average classification accuracy along with 95% confidence interval obtained for 3-fold stratified cross validation. The 3-fold cross validation is based on the three splits defined in Appendix A.2.10 of [75]. The three splits are balanced with respect to the two schools, the gender of the children and the four emotion classes. Thus in each fold, the training set

includes around 3000 chunks (speech utterances) and the test set includes around 1500 chunks.

Database for speaker identification from speech: Speaker identification experiments are performed on the 2002 and 2003 NIST speaker recognition (SRE) corpora [76, 77]. We considered the 122 male speakers that are common to the 2002 and 2003 NIST SRE corpora. Training data for a speaker includes a total of about 3 minutes of speech from the single conversations in the training set of 2002 and 2003 NIST SRE corpora. The test data from the 2003 NIST SRE corpus is used for testing the speaker recognition systems. Each test utterance is around 30 seconds long. The silence portions in the speech utterances are removed. Each utterance in the training and test sets is divided into segments of around 5 seconds. Each speech segment is considered as an example. This leads to a total of 3617 training examples with each speaker class having about 30 examples. The test set includes a total of 3044 examples. The speaker identification accuracy presented is the classification accuracy obtained for 3044 test examples. The training and test datasets as defined in the NIST SRE corpora are used in studies. Therefore, multi-fold cross-validation is not considered in the study on speaker identification.

Features for emotion recognition and speaker identification from speech: We have considered Mel frequency cepstral coefficients (MFCC) as features for emotion recognition and speaker identification from speech. A frame size of 20 ms and a shift of 10 ms are used for feature extraction from the speech signal of an utterance. Every frame is represented using a 39-dimensional feature vector. Here, the first 12 features are Mel frequency cepstral coefficients and the 13th feature is log energy. The remaining 26 features are the delta and acceleration coefficients. Thus, the speech signal of an utterance is represented as a sequence of 39-dimensional local feature vectors with each local feature vector representing a frame of the speech signal. The effectiveness of MFCC features for speech emotion recognition is shown in [78] and [75]. The effectiveness of MFCC features for speaker recognition is shown in [36].

13.7.1.2. Experimental studies on speech emotion recognition and speaker identification

The different GMM-based systems considered in the study include the GMMs whose parameters are estimated using the maximum likelihood (ML) method (MLGMM), large margin GMM (LMGMM) [79] and adapted GMM [69]. For GMM-based systems, the diagonal covariance matrices are considered. An MLGMM is built for each class with the number of components Q taking the value from the set $\{16, 32, 64, 128, 256\}$ for speech emotion recognition and speaker identification tasks. The large margin GMM-based system is built by refining the parameters of the MLGMM for each class using the large margin technique. In the large margin GMM method, the parameters of the GMMs for all the classes are estimated simultaneously by solving an optimization problem to maximize the distance of training examples to the boundaries of the classes. When the number of classes is large, the optimization problem solving in the large margin method for GMMs is computationally highly intensive. Therefore, the large margin GMM-based system is not considered in the studies on speaker identification. The adapted GMM-based system uses a class-independent GMM (CIGMM) with number of components Q taking the value from the set $\{256, 512, 1024, 2048\}$ for both speech emotion recognition and speaker identification tasks. The class-specific adapted GMMs are built by adapting the means, variances, and mixture coefficients of CIGMM to the training data of each class. A relevance factor of 16 is considered for adaptation as given in [69].

For dynamic kernel based SVM classifiers, we use the LIBSVM [80] tool to build the SVM classifiers. In this study, the one-against-the-rest approach is considered for 7-class and 4-class speech emotion recognition tasks and 122-class speaker identification task. The SVM classifiers are trained using different values of trade-off parameter, \mathfrak{C} . The best performance on test data is observed for $\mathfrak{C}=10$. For the Fisher kernel (FK) based SVM, class-wise GMMs with 64, 128, and 256 components are considered. In each case, the Fisher score vector is obtained for an example to compute Fisher kernel. For the probabilistic sequence kernel (PSK) based SVM, the CIGMMs with 256, 512, and 1024 components are considered. The CIGMM is adapted with the data of each speech emotion class or speaker to get the class-specific adapted GMM. For the GMM supervector kernel (GMMSVK) based SVM and the GMM-UBM mean interval kernel (GUMIK) based SVM, CIGMMs with 256, 512 and 1024 components are con-

sidered and an example-specific GMM is obtained by adapting the CIGMM to each of the examples. The IMK-based SVMs are built using a value of 128, 256, 512, 1024 and 2048 for Q corresponding to the size of set of virtual feature vectors.

The classification accuracies obtained for the MLGMM-based system, the large margin GMM-based system, the adapted GMM-based system and the SVM-based systems using FK, PSK, GMMSVK, GUMIK, summation kernel (SK), codebook-based IMK (CBIMK) and CIGMM-based IMK (CIGMMIMK) for both speech emotion recognition and speaker identification tasks are given in Table 13.1. The accuracies presented in Table 13.1 are the best accuracies observed among the GMM-based systems and SVMs with dynamic kernels using different values for Q . It is seen that the large margin GMM-based system gives a better performance than the MLGMM-based system. It is also seen that the adapted GMM-based system gives a better performance than the MLGMM-based and LMGMM-based systems for speech emotion recognition from Emo-DB and for speaker identification. The better performance of the adapted GMM-based system is mainly due to robust estimation of parameters using the limited amount of training data available for an emotion class, as explained in [69]. It is seen that the performance of adapted GMM-based system for speech emotion recognition from FAU-AEC is marginally better than that of MLGMM-based system. It is also seen that performance of the state-of-the-art dynamic kernel based SVMs is better than that of the GMM-based systems. This is mainly because the GMM-based classifier is trained using a non-discriminative learning based technique, where as the dynamic kernel based SVM classifier is built using a discriminative learning based technique. It is also seen that the performance of GUMIK-based SVMs is better than that of the SVM classifiers using FK, PSK, GMMSVK, SK, CBIMK and CIGMMIMK for speech emotion recognition from Emo-DB and for speaker identification. For the speech emotion recognition from FAU-AEC, the SVM using PSK performed better than the SVMs using FK, GMMSVK, GUMIK, SK and CBIMK, and marginally better than CIGMMIMK.

13.7.2. Scene image classification using dynamic kernel based SVMs

Scene image classification is not a trivial task because scene images are complex in nature. Multiple semantic objects occur in unspecified regions of a scene image. An important issue in semantic scene classification is intra-

Table 13.1. Comparison of classification accuracy (CA) (in %) of the GMM-based classifiers and SVM-based classifiers using FK, PSK, GMMSVK and GUMIK for speech emotion recognition (SER) task and speaker identification (Spk-ID) task. CA95%CI indicates average classification accuracy along with 95% confidence interval. Q indicates the number of components considered in building GMM for each class or the number of components considered in building CIGMM or the number of virtual feature vectors considered. NSV indicates the average number of support vectors.

Classification model	SER						Spk-ID			
	Emo-DB			FAU-AEC						
	Q	CA95%CI	NSV	Q	CA95%CI	NSV	Q	CA	NSV	
MLGMM	32	66.81±0.44	-	128	60.00±0.13	-	64	76.50	-	
LMGMM	32	72.71±0.43	-	128	61.03±0.14	-	-	-	-	
Adapted GMM	512	79.48±0.31	-	1024	61.09±0.12	-	1024	83.08	-	
SVM using	FK	256	87.05±0.24	184	512	61.54±0.11	1964	512	88.54	261
	PSK	1024	87.46±0.23	182	512	62.54±0.13	1873	1024	86.18	182
	GMMSVK	256	87.18±0.29	272	1024	59.78±0.19	2026	512	87.93	396
	GUMIK	256	88.17±0.34	298	1024	60.66±0.10	2227	512	90.31	387
	SK	-	75.09±0.28	104	-	56.06±0.09	1416	-	76.77	204
	CBIMK	512	80.54±0.27	223	1024	58.40±0.09	2185	1024	80.42	292
	CIGMMIMK	512	85.62±0.29	217	1024	62.48±0.07	2054	1024	88.54	261

class variability that corresponds to the extent of dissimilarity among the images of a semantic class. Another important issue in scene image classification is inter-class similarity that corresponds to the similarity among the images of different semantic classes. There are two approaches to classification of scene images depending on the way the set of local feature vectors is handled. In the first approach, a suitable method for classification of vectorial representation of scene images is used by first mapping a set of local feature vector representation of an image onto a fixed dimensional representation such as a supervector or a bag-of-visual-words (BOVW) representation. The BOVW representation of a scene image is a fixed dimensional vector of the frequencies of occurrence of visual words in a scene image. A limitation of BOVW representation is that there is loss of information. In the second approach to classification of scene images, a suitable method for classification of sets of local feature vectors representation of examples is explored.

In this Section, we present our studies on scene classification using GMM-based and SVM-based classifiers.

13.7.2.1. *Datasets and features used in the studies*

The experiments on scene image classification are performed on the following datasets. A summary of details of the datasets used in our studies on scene image classification is given in Table 13.2.

- *Vogel-Schiele (VS) dataset:* This dataset comprises of 700 scene images of 6 semantic classes, namely, ‘coasts’, ‘forests’, ‘mountains’, ‘open-country’, ‘river’ and ‘sky-clouds’ [34, 81]. The results presented for the studies using this dataset correspond to the average classification accuracy for 5-fold evaluation by considering 80% images from each class for training and the remaining for testing.
- *MIT-8-scene dataset:* The MIT-8-scene dataset comprises of 2688 scene images belonging to 8 semantic classes, namely, ‘coasts’, ‘forest’, ‘mountain’, ‘open-country’, ‘highway’, ‘inside-city’, ‘tall building’ and ‘street’ [82]. The results presented for the studies using this dataset correspond to the average classification accuracy for 10-fold evaluation by considering randomly chosen 150 images from each class for training.
- *Corel5K dataset:* Corel5K dataset comprises of images of 50 semantic classes where every class consists of 100 images [83]. The 50 classes in this dataset have a large amount of inter-class sim-

ilarity. The classes themselves have a high degree of intra-class variability. The results presented for the studies using this dataset correspond to the average classification accuracy for 5-fold evaluation by considering 80% of images from each class for training and the remaining for testing.

Table 13.2. Summary of details of datasets considered for the studies on scene image classification.

	Vogel-Schiele	MIT-8-scene	Corel5K
Number of classes	6	8	50
Total number of images	700	2680	5000
Number of images for testing	152	1488	1000
Number of folds	5	10	5
Size of an image in pixels	480x720	256x256	117x181
Number of blocks for an image	100 non-overlapped	36 non-overlapped	36 overlapped
Block representation	22-dimensional	22-dimensional	22-dimensional

Number of blocks considered for extraction of local feature vectors from an image in each dataset and the dimension of feature vector representing a block are given in Table 13.2. Each block in an image is represented by a 22-dimensional local feature vector comprising of HSV color histogram, edge direction histogram and wavelet based texture features. The color histogram is computed separately for hue (4 bins), value (2 bins) and saturation (2 bins) to obtain 8 elements in the local feature vector. The edge direction histogram is computed using 8 bins of 45 degrees apart that contributes 8 elements in the local feature vector. The remaining 6 elements in a 22-dimensional local feature vector are obtained using wavelet based texture features.

13.7.2.2. Studies on scene image classification using GMM-based and SVM-based methods

We consider the GMM-based methods and the SVM-based methods for scene image classification. In the GMM-based methods, we consider ML-GMMs and LMGMMs. The best value for number of components (Q) for the GMM of a class is chosen empirically from among 8, 16, 32, 64, 128, 256, and 512 based on classification accuracy. For SVM-based classifiers, we consider the supervector representation, BOVW representation and sets of

local feature vectors representation of scene images. Local feature vectors of an image are concatenated into a supervector and a Gaussian kernel (GK) is used for the SVM-based classification. Suitable value for the width of the GK is chosen empirically. For the BOVW representation, we consider the K -means clustering based method and the GMM-based method. We select the best value for the number of visual words (NVW) from among 8, 16, 32, 64, 128, 256, 512, and 1024 based on classification accuracy. We consider GK and histogram intersection kernel (HIK) for SVM-based classification of scene images represented in the BOVW form. For the set of local feature vectors representation of a scene image, we consider GMMSVK, GUMIK and IMK. To construct GMMSVK, GUMIK and IMK, we build an UBM considering the training data of all the classes. The best number of components for UBM, Q , is chosen empirically from among 8, 16, 32, 64, 128, 256, 512, and 1024 by a method of line search based on classification accuracy. The width parameter of the GK used as a base kernel in IMK is chosen empirically. The UBM is adapted to the data of every example to construct the GMMSVK and GUMIK. The performance of SVM-based classifiers is compared with that of GMM-based classifiers in Table 13.3. The accuracy presented in Table 13.3 is the best accuracy observed among GMM-based classifiers and SVM-based classifiers using different values for Q or NVW.

Table 13.3. Average classification accuracy (CA)(in %) over multiple folds for GMM-based and SVM-based classifiers for scene image classification. Here, Q corresponds to the number of components of the GMM of a class in the GMM-based classifier and that of the UBM used in construction of IMK, GMMSVK and GUMIK in the SVM-based classifiers. NVW correspond to the number of visual words chosen for the BOVW representation of a scene image.

Representation	Method for classification	VS dataset		MIT-8-scene dataset		Corel5K dataset	
		$Q/$ NVW	CA	$Q/$ NVW	CA	$Q/$ NVW	CA
Supervector	GK-SVM	-	58.29	-	75.85	-	38.00
BOVW (K -means)	GK-SVM	128	48.42	64	47.46	256	24.24
	HIK-SVM	512	44.08	256	37.42	256	15.52
BOVW (GMM)	GK-SVM	64	53.95	256	63.65	256	27.60
	GK-SVM	256	49.21	256	55.69	256	20.06
Set of local feature vectors	MLGMM	64	49.21	256	63.16	128	36.58
	LMGMM	64	49.34	256	63.14	128	36.66
	GMMSVK-SVM	8	51.97	8	62.34	8	25.92
	GUMIK-SVM	8	55.53	8	64.01	16	30.94
	IMK-SVM	256	57.63	256	71.72	256	40.52

It is seen from Table 13.3 that the SVM-based methods for scene image classification using the dynamic kernels on the sets of local feature vectors representation and Gaussian kernel on the supervector represen-

tation perform better than the GMM-based methods and the SVM-based classifiers using the BOVW representation of scene images. It is also seen that there is only a marginal improvement in classification accuracy when LMGMM is considered. Among the SVM-based methods, it is seen that the GK-based SVM using supervector representation performs better than the other methods for the VS dataset and MIT-8-scene dataset. However, it is also seen that the IMK-based SVMs perform close to or better than the GK-based SVMs using supervector representation. The performance of the SVMs considering the supervector is expected to be better as all the local feature vectors participate in the computation of the kernel function. It is interesting to see that the IMK-based SVM performs close to or better than the supervector based approach, though only a subset of local feature vectors are considered. However, the SVM-based method using a dynamic kernel can also be used when the varying length set of local feature vectors are used for representing a scene image. It is also seen that the approach of mapping a set of local feature vectors onto a BOVW representation and then using it for classification does not perform well.

13.7.3. Kernel based matching for content based image retrieval of scene images

Matching a pair of scene images for content based image retrieval (CBIR) of scene images is presented in this section. In CBIR, all the images in the repository that are similar to the query image presented by the user are to be retrieved. An important issue in CBIR of scene images using the Query-by-Example (QBE) method is judging the relevance of images in the repository to a query image. The QBE method requires a suitable approach to match a pair of scene images.

Approaches to match a pair of scene images include dissimilarity based matching and similarity based matching. There are two approaches to match a pair of scene images based on the way in which the set of local feature vectors representation of a scene image is handled. In the first approach, a suitable method for matching vectorial representations of scene images is used by first mapping a set of local feature vectors corresponding to an image onto a fixed dimensional vector representation such as a BOVW representation. The supervector representation of a set of local feature vectors is suitable only when the number of local feature vectors is the same for all the scene images. Distance measures such as Euclidean distance [14, 15, 31, 84], L_1 norm [16, 19, 85] and Mahalanobis distance [86] can

be used in the dissimilarity based matching of two scene images represented as fixed dimensional vectors. The BOVW representation corresponds to a discrete probability distribution of visual words. The Kullback-Leibler divergence [87] or Jensen-Shannon divergence [87, 88] can be used as a measure of distance between two scene images represented in the BOVW form. A measure of similarity such as cosine similarity or histogram intersection function can be used in the similarity based matching of a pair of scene images represented in BOVW form. A limitation of considering the BOVW representation of a scene image is that there is a loss of information due to quantization process involved in obtaining it. These limitations are addressed in the second approach to matching of scene images where a suitable method for matching the sets of local feature vectors corresponding to scene images is used. Distance measures such as earth mover's distance [89, 90] and C_2 divergence [91] between the models of probability distributions corresponding to sets of local features vectors of two scene images are used in the dissimilarity based matching. The focus of this section is on the kernel based matching as a similarity based matching method to be used for CBIR of scene images using the QBE method.

In the kernel based matching method, the value of a kernel function is used as a measure of similarity between a pair of scene images. An important issue in the kernel based matching is the selection of a suitable kernel for the chosen representation of scene images. We consider linear kernel, Gaussian kernel, polynomial kernel, Bhattacharyya kernel, histogram intersection kernel [92, 93], bag-of-words kernel [94, 95] and term frequency log likelihood ratio kernel [96] in the similarity based matching of a pair of scene images represented in the BOVW form. Kernels for the sets of local feature vectors representations of examples are known as dynamic kernels [97]. We consider three dynamic kernels, namely, intermediate matching kernel (IMK) [98], GMM supervector kernel (GMMSVK) [4] and GMM universal background model mean interval kernel (GUMIK) [59] for matching two images represented as sets of local feature vectors.

13.7.3.1. *Datasets and features used in the studies*

Availability of a suitable benchmark dataset is important for the evaluation of the performance of a CBIR system. A benchmark dataset for retrieval should comprise of a repository of images, a test suite of query images and a set of relevance judgements for the test suite of query images [99]. The relevance judgements are often binary in nature indicating whether

an image in the repository is relevant for a particular query image or not. The set of relevant images for a query is known as ground truth or gold standard. To evaluate a CBIR system using the QBE method, the queries are presented to the system. For every query image, the images retrieved by the CBIR system are compared with the images in its ground truth to judge which of the retrieved images are indeed relevant. The studies on CBIR using the kernel based matching are performed on the following datasets.

- *Uncompressed color image database (UCID)*: It comprises of 1338 images, corresponding to a variety of semantic content including man-made objects to natural scenes [100]. The images correspond to indoor scenes as well as outdoor scenes. The dataset also comprises of ground truth of relevant images for each image in a subset of 262 images. The number of the relevant images in the ground truth for every query image is different. In this study, we consider only 77 query images, each of which has at least 3 relevant images in its ground truth. It is seen that the dataset has high variability in content and the retrieval task on this dataset is a difficult one due to the variability in the content [101].
- *University of Washington (UW) dataset*: The images in this dataset include pictures from various locations that are manually annotated by human experts [101]. We consider 60% of the images in the dataset as the retrieval repository and 40% of the images as query images. The results presented correspond to the average of the performance for 5-fold evaluation.
- *MIT-8-scene dataset*: MIT-8-scene dataset comprises of 2688 scene images belonging to 8 semantic classes, namely, ‘coasts’, ‘forest’, ‘mountain’, ‘open-country’, ‘highway’, ‘inside-city’, ‘tallbuilding’, and ‘street’ [82]. We consider 150 images of each class for inclusion in the retrieval repository. The remaining images are considered as query images. The results presented correspond to the average performance for 10-fold evaluation.
- *Corel5K dataset*: Corel5K dataset comprises of images of 50 semantic classes where every class consists of 100 images [83]. The 50 classes in this dataset have a large amount of inter-class similarity. There is also a high degree of intra-class variability. From every class, 20 images are considered as query images and the remaining images in each class are pooled into retrieval repository.

The results presented correspond to the average performance for 5-fold evaluation.

A summary of details of the datasets used in our studies on image retrieval is given in Table 13.4. The details of the blocks considered for the extraction of local feature vectors and dimension of a local feature vector extracted from a block of an image in different datasets are given in Table 13.4. Each block of an image is represented using a local feature vector comprising of color histogram, edge direction histogram and wavelet transform based texture features. The details of 22-dimensional block representation is already provided in Section 13.7.2.1. The 30-dimensional vector representation of a block includes 8 bins for hue and 4 bins each for saturation and value. The 38-dimensional vector representation of a block includes 16 bins for hue and 4 bins each for saturation and value. The histograms for hue, saturation and value are concatenated to obtain the color histogram representation. The 30-dimensional vector representation of a block and 38-dimensional vector representation of a block considered for images in UCID are called as Representation 1 and Representation 2 respectively.

Table 13.4. Summary of details of datasets considered for the studies on CBIR using the kernel based matching.

	UCID	UW	MIT-8-scene	Corel5K
Number of images	1338	1109	2680	5000
Number of query images	1267	446	1480	1000
Number of folds	1	5	10	5
Size of an image	384×512	Different size	256×256	117×181
Number of blocks for an image	100 overlapped	225 non-overlapped	36 non-overlapped	36 overlapped
Block representation	30-dimensional 38-dimensional	22-dimensional	22-dimensional	22-dimensional
Ground truth for retrieval	Available	Not available	Not available	Not available

The performance of a CBIR system is evaluated using the precision-recall curves and the mean average precision (MAP) score [99]. Ground truth for retrieval is not available for MIT-8-scene dataset, Corel5K dataset and UW dataset. For a query image in MIT-8-scene dataset and Corel5K dataset, all the images in its respective semantic class are considered as its

ground truth. Images in the UW dataset comprise of concept labels which are used to derive the ground truth for retrieval as presented here. Let G_i be the set of concept labels in the annotation ground truth for an image I_i . Let \mathcal{D}^c be the set of images that have the concept c in their annotation ground truth. The image I_i is in \mathcal{D}^c provided c is in G_i . Let G_q be the set of concept labels in the annotation ground truth for the query image I_q . Now, the set of images, R_q , relevant for I_q is given by [101]

$$R_q = \bigcup_{c \in G_q} \mathcal{D}^c \quad (13.91)$$

13.7.3.2. Studies on matching the vectorial representations of scene images

The vectorial representations considered in this study are the supervector representation and the BOVW representation. For the supervector representation of the scene images, Euclidean distance(ED) is used as a measure of dissimilarity for matching. For BOVW representation, the K -means clustering based method and GMM-based method to obtain visual words are considered. We consider the best value for the number of visual words (NVW) from among 32, 64, 128, 256, 512, and 1024. Here, the best value for NVW is chosen based on the best value for the MAP score obtained for CBIR involving the respective BOVW representation of scene images. For the BOVW representation of scene images, ED, KL divergence (KLD) and Jensen-Shannon divergence (JSD) are considered for dissimilarity based matching. For the similarity based matching of BOVW representation of scene images, the histogram intersection kernel (HIK), bag-of-words kernel (BOWK) and term frequency log likelihood ratio kernel (TFLLRK) are used. In Figure 13.15 the MAP scores for CBIR of scene images of UCID dataset using ED, KLD, JSD, HIK, BOWK, and TFLLRK are compared. It is seen from Figure 13.15 that MAP score for HIK based matching is better than the other methods. In Figure 13.16, we compare the MAP scores obtained for BOVW representation obtained using the K -means clustering method and the GMM-based method for the CBIR of scene images in different datasets. It is seen that the MAP score for the BOVW representation, obtained using the GMM-based method is marginally better than that obtained using the K -means clustering method.

We compare the MAP scores for CBIR using the supervector representation and BOVWG representation of scene images for the different datasets in Figure 13.17. Here, ED is used for matching the supervector represen-

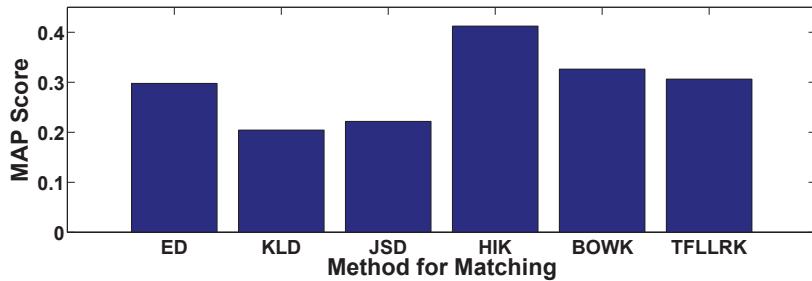


Fig. 13.15. Comparison of MAP scores for CBIR of scene images using different methods for matching the BOVW representation of the images in UCID.

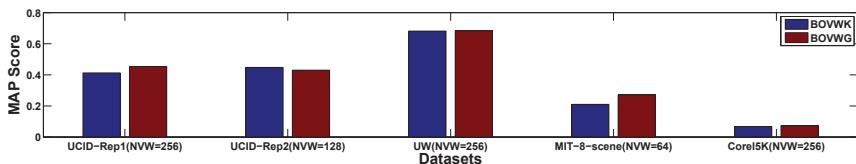


Fig. 13.16. Comparison of MAP scores for CBIR of scene images in different datasets by matching the BOVW representations of images obtained using K -means clustering method (BOVWK) and GMM-based method (BOVWG).

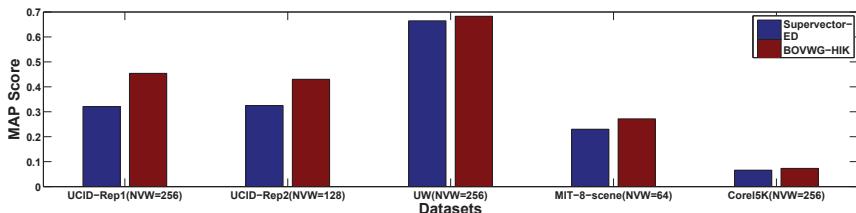


Fig. 13.17. Comparison of MAP scores for CBIR of scene images in different datasets by matching the supervector representations of images using Euclidean distance and by matching the BOVWG representations of images using histogram intersection kernel.

tation and HIK is used for matching the BOVWG representation of scene images. It is seen from Figure 13.17 that MAP score for CBIR for matching the BOVW representations of scene images is better than that for CBIR by matching the supervector representations of images in UCID.

13.7.3.3. Studies on matching sets of local feature vectors representation

In this section, we present studies on matching the sets of local feature vectors using C_2 divergence and dynamic kernels. A GMM is built for every scene image as a model of the probability distribution of the local feature vectors of it. The number of components for a GMM (Q) of an image is selected empirically based on the best value of MAP score for the CBIR using it. For similarity based matching, we consider three dynamic kernels, namely, IMK, GMMSVK, and GUMIK that are constructed using a universal background model (UBM). The UBM is built using the local feature vectors of all the scene images in the retrieval repository. The best number of components (Q) of the UBM is chosen empirically from among 32, 64, 128, 256, 512, and 1024 based on MAP scores for matching using the kernels computed using the UBM. Gaussian kernel (GK) is used as the base kernel in the construction of IMK. The width parameter of GK is chosen empirically by a method of line search. We adapt UBM to the data of every scene image to compute GMMSVK and GUMIK. The performance of CBIR using the set of local feature vectors representations of scene images in UCID is compared with that for matching the supervector and BOVWG representation of scene images using precision-recall curves in Figure 13.18. It is seen from Figure 13.18 that the kernel based matching for fixed dimensional vector representation and the set of local feature vectors representation performs better than that for matching using ED on supervector representation and C_2 divergence based matching on the

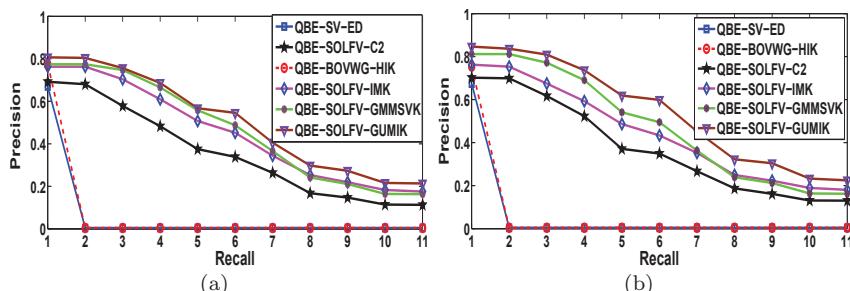


Fig. 13.18. The precision-recall curves for the QBE method of CBIR of scene images using different approaches to matching for (a) 30-dimensional local feature vector representations of blocks and (b) 38-dimensional local feature vector representations of blocks of images in UCID.

sets of local feature vectors representation of images in UCID. Among the different dynamic kernels considered, CBIR involving matching the sets of local feature vectors using GUMIK performs better.

In Figure 13.19, precision-recall curves to compare different approaches to matching images in UW dataset are given. In Figure 13.20, precision-recall curves to compare different approaches to matching images in MIT-8-scene dataset are given. It is seen from Figure 13.19 that for images in UW dataset, kernel based matching on BOVW representation performs

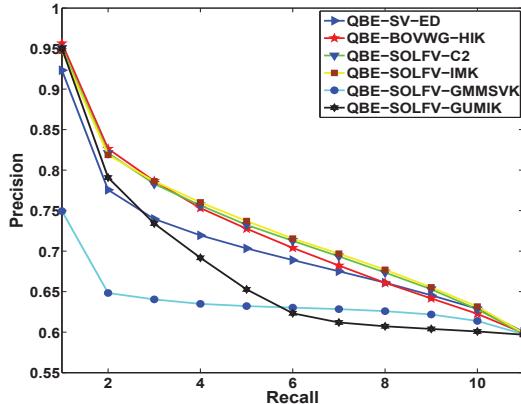


Fig. 13.19. The precision-recall curves for the QBE method of CBIR of scene images using different approaches to matching of images in UW dataset.

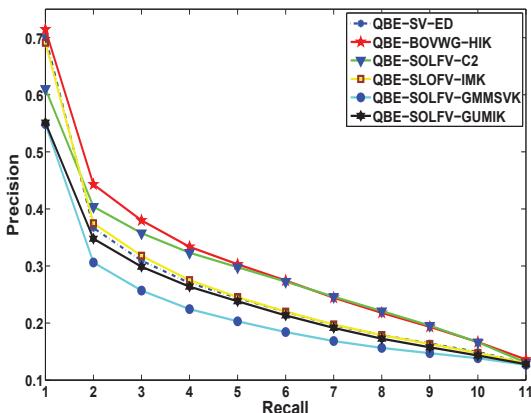


Fig. 13.20. The precision-recall curves for the QBE method of CBIR of scene images using different approaches to matching of images in MIT-8-scene dataset.

better than $C2$ divergence based matching. It is also seen that dynamic kernel based matching of set of local feature vectors representation performs better than kernel based matching on BOVW representation. For images in UW-dataset, IMK-based matching performs better than the remaining approaches to matching. It is also seen from Figure 13.20 that IMK-based matching of images in MIT-8-scene dataset performs better than the other dynamic kernels. However, it is seen that for MIT-8-scene dataset, $C2$ divergence based matching using GMMs performs close to that of HIK based matching on the BOVW representation of scene images. The better performance of $C2$ divergence based matching may be due to the nature of ground truth in MIT-8-scene dataset. This dataset is a benchmark dataset for classification task that has semantic classes that are more coherent. So, while matching also, this might have been helpful. It is also seen that for images in MIT-8-scene dataset, using HIK-based matching on BOVW representation performs better than the other approaches. It is also seen that for images in Corel5K, matching using IMK performs better than the other approaches.

13.7.4. Scene image annotation using dynamic kernel based methods

Scene image annotation involves assigning textual concept labels to a scene image based on its semantic content. Automatic scene image annotation involves bridging the semantic gap between the low-level image visual features and the high-level semantic concepts. Bridging the semantic gap is not a trivial task as there is no explicit characterization of the semantic gap due to the subjectivity in the perception of the semantic content. The approaches to develop an automatic image annotation system include the non-model based approaches and the model based approaches. The non-model based approaches are based on the principle of propagation of labels from the scene images in the training dataset that are similar to a test image [102–104]. The nearest neighbor label propagation (NNLP) method for annotation involves a two-stage label transfer algorithm [102]. In the first stage, the concept labels from the first nearest neighbor are transferred. If the first nearest neighbor has more than M concept labels, their frequency of occurrence in the training dataset is considered for rank ordering of the concept labels. If the first nearest neighbor has less than M concept labels, the concept labels from the second through K th nearest neighbors are transferred in the second stage of the label transfer, where co-occurrence

factor and local frequency factors of concept labels are considered for rank ordering them. The co-occurrence factor considers how many times the concept labels that are already assigned to I co-occur with the concept labels of the second through K th nearest neighbors in the training dataset. The local frequency factor of a concept label corresponds to the frequency of occurrence of a concept label in the second through K th nearest neighbors of I .

The model based approaches involve building models to bridge the semantic gap between the low-level image features and the semantic concepts. The models may be built using either unsupervised learning based approaches [105] or supervised learning based approaches [106]. In the unsupervised learning based approaches, co-occurrence of the image visual features and semantic concept labels is modelled. In the supervised learning based approaches, a model is built for every semantic concept and a test image is annotated using the outputs of the models for the semantic concepts. An important issue in the supervised learning based approaches to image annotation is related to the multi-label nature of scene image data, because a scene image typically comprises of multiple concept labels. In such a case, a scene image in the training dataset has to be used as a positive example in building the models for the concepts corresponding to all the labels in its ground truth. In this work, we focus on the supervised learning based approaches to annotation of scene images represented as sets of local feature vectors.

Conventionally, Gaussian mixture model (GMM) based method that involves building a GMM for every semantic concept [106] is used for annotating a scene image represented as sets of local feature vectors. A test image is annotated using the posterior probabilities corresponding to all the semantic concepts. We present a method for decision logic that uses a threshold \mathcal{T}_i on the posterior probabilities for a scene image I_i obtained as follows:

$$\mathcal{T}_i = \frac{P_{imax}}{\mathcal{C}} \quad (13.92)$$

Here P_{imax} is the maximum of the posterior probabilities of concepts for the image I_i and \mathcal{C} is the total number of concepts. The concept labels corresponding to all those concepts for which the posterior probabilities are greater than \mathcal{T}_i are used for annotating the image I_i . A limitation of this method is the need for building a model of the probability distribution of every concept using the local feature vectors of all the images having that concept in their ground truth. Due to the multi-label nature of data, the

local feature vectors corresponding to the other concepts also participate in building the model for a concept. This limitation is addressed in the SVM-based method for annotation where there is no need to build models of probability distribution for concepts. In this method, one SVM is built for every semantic concept to discriminate the images having that concept label in their ground truth from all the images that do not have that concept label in their ground truth.

An important issue in the SVM-based method for annotation is related to training the concept SVMs with the multi-label data. A scene image with multiple labels becomes a positive example for all those concept SVMs corresponding to the concept labels in its ground truth. We use a method of cross-training of the concept SVMs [107, 108]. In this method of training, images with a concept label in their ground truth are considered as the positive examples for the corresponding concept SVM. All the images which do not have that concept label in their ground truth are considered as the negative examples. A scene image may be used as a positive example for more than one concept SVM. The output score, s_c of concept SVM corresponding to concept w_c is mapped onto a pseudo probability using a logistic function as follows [109]:

$$\tilde{P}(w_c|\mathbf{X}) = \frac{1}{1 + \exp(-\alpha s_c)} \quad (13.93)$$

where α is a free parameter that controls the slope of the logistic function. The pseudo probability value is normalized to obtain the posterior probability value corresponding to a concept as follows:

$$P(w_c|\mathbf{X}) = \frac{\tilde{P}(w_c|\mathbf{X})}{\sum_{i=1}^c \tilde{P}(w_i|\mathbf{X})} \quad (13.94)$$

In the following sections, we present our studies on scene image annotation using GMM-based and SVM-based methods.

13.7.4.1. Datasets and features used in the studies

The experimental studies on scene image annotation are performed on the following datasets in this section:

- University of Washington (UW) annotation benchmark dataset: The images in this dataset include pictures from various locations that are manually annotated by human experts [101, 110]. All similar concept labels which when considered separately have a very

low frequency of occurrence in the dataset are merged manually into a single concept label. Among them, the frequency of occurrence of a few concepts is very small. We consider the 57 concepts that have a minimum of 25 as the frequency of occurrence. We consider 60% of the images in the dataset for training the concept models and 40% of the images for testing. The results presented correspond to the average of annotation performance for repeated random sub-sampling validation for 5 folds.

- IAPR TC-12 dataset: This is a dataset of about 20000 images of natural scenes [102]. The images in the dataset are accompanied by the free flowing textual description of the semantic content in three different languages, namely, English, German and Spanish. The English text associated with every image is processed to extract keywords that are made available by the authors of [102]. The results presented for this dataset correspond to the performance of the single fold of train and test set as given in [102].

Every scene image is represented by a set of local feature vectors where a local feature vector is extracted from a block of size 50×30 pixels of scene image. Each block is represented by a 22-dimensional local feature vector whose details are already presented in Section 13.7.2.1. The images in UW dataset and IAPR TC-12 datasets are of different sizes. This results in varying length set of local feature vectors representation of scene images.

We consider mean precision (\tilde{P}) and mean recall (\tilde{R}) as the performance measures. The mean precision corresponds to the mean of the per concept precision (P_c) and the mean recall is the mean of the per concept recall (R_c). The precision for a concept c is given by $P_c = \frac{L_{cc}}{L_{ac}}$, where L_{cc} denotes the number of images that are correctly annotated with the concept c and L_{ac} is the number of images that are annotated with the concept c by an annotation method. The per concept recall is given by $R_c = \frac{L_{cc}}{L_c}$, where L_c denotes the number of images in the test set that contain the concept c in their ground truth. The precision and recall trade off against each other. An increase in recall is associated with a decrease in precision. We consider a single measure known as F measure that corresponds to the harmonic mean of precision and recall computed as follows [99]:

$$F = \frac{2\tilde{P}\tilde{R}}{\tilde{P} + \tilde{R}} \quad (13.95)$$

It is also a normal practice to evaluate an annotation method by studying the performance of a retrieval system into which it is integrated. For this

study, we consider the CBIR system using the Query-by-Semantics (QBS) method where single word queries are used. The QBS method involves retrieval of all those images in the repository that have the concept label that is same as the query word. We consider the precision-recall curves for evaluating the retrieval performance of the CBIR system using the QBS method.

13.7.4.2. Studies using UW annotation benchmark dataset

Study on methods for annotation: We study the SVM-based method, GMM-based method and the nearest neighbor label propagation (NNLP) based method for annotation. We consider three dynamic kernels, namely, IMK, GMMSVK, and GUMIK for the SVM-based methods for annotation for set of local feature vectors representation of a scene image. The IMK is computed using the components of UBM as the virtual feature vectors. The width parameter of the Gaussian kernel is chosen empirically by a method of line search. The UBM is adapted to the data of every scene image to compute the GMMSVK and the GUMIK. The optimal number of components for the UBM is chosen empirically from among 16, 32, 64, 128, 256, and 512 by a method of line search based on the F measure obtained using the SVM-based method for annotation that considers the respective dynamic kernel. We also consider a combined dynamic kernel that corresponds to the sum of IMK, GMMSVK, and GUMIK for a given scene image representation. The number of components for the concept GMMs in the GMM-based method for annotation is also chosen empirically from among 8, 16, 32, 64, 128, 256, and 512 components in the method of line search based on F measure. In our study, we consider the kernel feature space for the NNLP based method for annotation where the value of the kernel is considered as a measure of similarity to identify the nearest neighbors of the test image. An important issue in this method of annotation is the selection of a suitable value for the neighborhood size, K , and M . Here, M is the number of concept labels that need to be propagated from the nearest neighbors to the test image. In our work, we empirically select a suitable value for K and M by a line search method so that an optimal combination of (K, M) is chosen that gives the maximum F measure. The performance of different methods for annotation is compared in Table 13.5. We compare the performance of the SVM-based method for annotation with that of the NNLP-based method in the respective kernel feature spaces in Figure 13.21.

Table 13.5. Mean precision (\tilde{P}), mean recall (\tilde{R}), and F measure (F) for different methods of scene image annotation. Here, Q corresponds to the number of components for each of the concept GMMs in the GMM based method for annotation and the number of components for the UBM used in the computation of IMK, GMMSVK and GUMIK. Here, (K, M) correspond to the size of the neighborhood and the number of concept labels propagated from the nearest neighbors. The combined dynamic kernel corresponds to the sum of IMK, GMMSVK and GUMIK.

Annotation method	Kernel	Q	(K, M)	\tilde{P}	\tilde{R}	F
NNLP	IMK	256	(50,15)	0.39608	0.68631	0.50137
	GMMSVK	64	(50,15)	0.45820	0.71988	0.55980
	GUMIK	128	(25,10)	0.50448	0.65917	0.57088
GMM	-	128	-	0.51441	0.56888	0.54013
SVM	IMK	256	-	0.58332	0.60361	0.59231
	GMMSVK	64	-	0.59475	0.57338	0.58366
	GUMIK	128	-	0.54990	0.65356	0.59700
	Combined dynamic kernel	-	-	0.64608	0.63448	0.61396

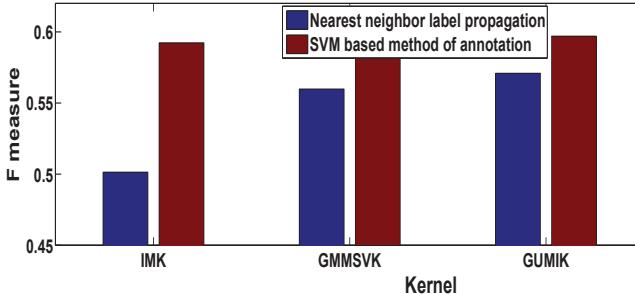


Fig. 13.21. Comparison of F measure for the nearest neighbor label propagation method and the SVM based method of annotation for different kernels, for the scene images in UW dataset. Here, the F measure is plotted from 0.45.

It is seen from Tables 13.5 and Figure 13.21 that the SVM classifier based methods perform better than the GMM-based method for annotation and the NNLP-based method. Among the dynamic kernel based SVM methods, the IMK and the GUMIK-based methods perform better than the GMMSVK-based methods. Computation of GMMSVK and GUMIK involves adapting the UBM to the data of every example. This is a computationally intensive process. The GUMIK-based SVMs give a better performance than the GMMSVK-based SVMs. The IMK-SVM-based method performs close to or marginally better than the GUMIK-SVM based method. The performance of the SVM-based method using the combined dynamic kernel is better than that of the other SVM-based methods. The performance for the varying length representation is comparable to that of the fixed length representation.

Study on CBIR of scene images using the QBS method: Next, we present the performance of CBIR using the QBS method that uses a single query word. The CBIR system makes use of the concept labels assigned by the model based annotation methods being studied. For this study, we consider the GMM-based annotation method and the SVM-based annotation methods that use dynamic kernels on the images represented using the varying length set of local feature vectors representation and the fixed length set of local feature vectors representation of scene images. As the dataset considered for annotation is not a retrieval benchmark dataset, the following procedure is used for creating the ground truth for the QBS method of retrieval. Let G_i be the set of concept labels in the annotation ground truth for an image I_i . Let \mathcal{D}^c be the set of images that have the concept label w_c in their annotation ground truth. Now, the image I_i is in \mathcal{D}^c provided w_c is in G_i . Given the concept labels assigned to each of the images by a particular method of annotation, all the images whose annotation includes the concept label corresponding to the query word are considered as relevant for that query. The methods for annotation use the posterior probabilities for the concepts to choose a suitable set of concept labels for the images. For a query word w_c , the images that are annotated with w_c by a method of annotation are rank ordered by the decreasing order of the value of $P(w_c|\mathbf{X})$. We consider the images in the test data of annotation models as the retrieval pool. The performance of CBIR of scene images with the QBS method using the labels assigned by the different methods of annotation is compared using the precision-recall curves given in Figure 13.22. It is seen that the retrieval performance using the concept labels assigned by the dynamic kernel based SVM classifier methods for annotation is better than that of the CBIR system using the concept labels assigned by the GMM-based method of annotation.

13.7.4.3. Study on annotation using IAPR TC-12 dataset

In this section, we present the results of our studies on annotation using the IAPR TC-12 dataset. We consider the same set of training and test images as used in [102]. The results of the annotation using the proposed method are compared with the state-of-the-art methods in Table 13.6. It is seen that the performance of annotation using the SVM-based methods is better than that for the nearest neighbor label propagation method and the GMM-based method. It is also seen that the performance of the methods using the dynamic kernels is better than that of the GMM-based method.

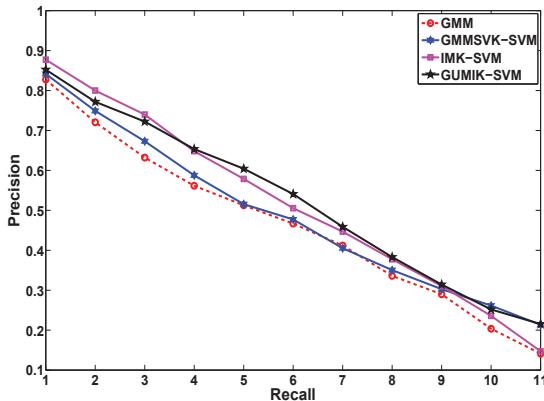


Fig. 13.22. The precision-recall curves for the QBS method of CBIR of scene images in UW dataset.

Table 13.6. Comparison of performance of different methods for annotation for IAPR TC-12 dataset.

Annotation Method	Kernel	Q	\bar{P}	\bar{R}	F
NNLP	IMK	1024	0.18723	0.10890	0.13771
GMM	-	128	0.21023	0.08076	0.11670
SVM	IMK	1024	0.35007	0.16820	0.22722
	GMMSVK	32	0.31708	0.15460	0.20786
	GUMIK	1024	0.31772	0.20746	0.24096

The precision for the GMM-based method is poorer than that for the SVM-based methods.

13.8. Studies on classification of varying length patterns represented as sequences of local feature vectors

In this Section, we present our studies on classification of isolated utterances of the E-set of English alphabet and recognition of CV units from the continuous speech corpus of Hindi using maximum likelihood HMM-based system, the large margin HMM-based system, and the SVM-based classifiers with dynamic kernels such as score space kernels, alignment kernels and matching based kernels.

13.8.1. Databases and features used in the studies

The proposed approaches to design the HMM-based IMKs are studied in building the SVM-based classifiers for different speech recognition tasks. The tasks include recognition of isolated utterances of the E-set of English alphabet and recognition of consonant-vowel (CV) segments Hindi. The Oregon Graduate Institute (OGI) spoken letter database is used in the study on recognition of E-set [111]. The continuous speech corpus of broadcast news in Hindi [112] is used for the study on recognition of CV segments.

13.8.1.1. Database for E-set recognition

The E-set is a highly confusable subset of spoken letters in English alphabet. The Oregon Graduate Institute (OGI) spoken letter database [111] includes the utterance data for 9 letters: B, C, D, E, G, P, T, V, and Z. The training data set consists of 240 utterances from 120 speakers for each letter, and the test data set consists of 60 utterances from 30 speakers for each letter. This leads to a total of 2160 training examples and a total of 540 test examples. The E-set recognition accuracy presented is the classification accuracy obtained for 540 test examples.

13.8.1.2. Database for recognition of CV segments

The experiments on recognition of CV segments are performed on continuous speech corpus of broadcast news in Hindi [112]. We considered 103 CV classes that have at least 50 examples in the training data set [112]. The data set consists of 19,458 CV segments for training and 4,866 CV segments for testing. The list of CV classes of Hindi considered in the studies on recognition of CV segments is given in Table 13.7. The CV segment recognition accuracy presented is the average classification accuracy along with 95% confidence interval obtained for 5-fold stratified cross-validation.

13.8.1.3. Features for E-set and CV segment recognition

We have considered Mel frequency cepstral coefficients (MFCC) as features for E-set recognition and CV segment recognition. A frame size of 25.6 ms and a shift of 10 ms are used for feature extraction from the speech signal of an utterance from E-set data (as given in [63]). A frame size of 20 ms and a shift of 10 ms are used for feature extraction from the speech signal of an utterance from CV segment data. Every frame is represented using a

Table 13.7. The number of examples (NoE) considered for each of CV classes of Hindi language.

Class	NoE	Class	NoE	Class	NoE	Class	NoE
ke	1533	vi	300	sha	140	kha	76
ne	1213	li	293	Thi	138	me	76
kI	766	II	275	di	136	DI	71
se	734	pa	265	pai	136	yo	67
yA	681	hI	261	sA	124	no	66
kA	637	re	260	dI	122	sau	66
hai	628	bhA	246	ri	110	je	64
ki	616	mA	243	la	108	da	62
ye	613	nI	242	sai	108	vo	62
ra	524	rA	242	dA	107	dhA	60
ko	522	ja	237	chA	104	go	60
tA	510	ni	233	TI	103	mI	60
nA	496	ha	232	khA	99	rU	60
ka	479	tI	232	do	99	shA	60
ya	446	ho	224	mu	95	DhA	58
sa	429	ma	218	TA	95	mi	58
rI	398	gA	190	he	93	bi	58
vA	394	ta	185	thA	93	DA	56
hA	394	hu	180	ve	91	the	56
pA	384	ti	180	gI	88	yI	56
na	373	bhI	178	ge	87	cha	55
ba	362	de	158	su	84	dhi	55
le	362	sI	154	jI	81	shu	55
jA	332	te	150	bu	79	nu	51
ga	316	bA	150	lo	79	ro	51
lA	314	va	143	ShA	77		

39-dimensional feature vector. Here, the first 12 features are Mel frequency cepstral coefficients and the 13th feature is log energy. The remaining 26 features are the delta and acceleration coefficients. The effectiveness of MFCC features for E-set recognition is shown in [63, 113]. The effectiveness of MFCC features for recognition of CV segments is shown in [112].

13.8.2. Experimental studies on E-set recognition and recognition of CV segments in Hindi

In this section, we first present results of our studies on E-set recognition and recognition of CV segments in Hindi using the HMM-based classifiers and the state-of-the-art dynamic kernel based SVM classifiers. Then we present the results of our studies using SVM classifiers with HIMKs and

the HIMKs using weighted base kernels.

The left-to-right continuous density HMM (CDHMM) based systems are considered in the study. The CDHMM parameters are estimated using the maximum likelihood (ML) method and the large margin (LM) method [114]. The diagonal covariance matrices are used for the state-specific GMMs. A CDHMM is built for each class with the number of states N taking the value from the set $\{5, 6, 7, 8, 9, 10, 11, 12\}$ and the number of components Q taking the value from the set $\{1, 2, 3, 4, 5\}$ for the GMM of each state for E-set recognition task. For recognition of CV segments in Hindi, a CDHMM is built for each class with the number of states N taking the value from the set $\{5, 6\}$ and the number of components Q taking the value from the set $\{1, 2, 3\}$ for the GMM of each state. The large margin HMM (LMHMM) based system is built by refining the parameters of the maximum likelihood HMM (MLHMM) for each class using the large margin technique. In the large margin HMM method, the parameters of the HMMs for all the classes are estimated simultaneously by solving an optimization problem to maximize the distance between the boundaries of each class with the rest of the classes.

For construction of score space kernels such as Fisher kernel (FK) and likelihood ratio kernel (LRK), the MLHMMs with the specifications as mentioned earlier are used. The Fisher score vector is obtained for each example to compute the Fisher kernel. We have implemented the FK and LRK as per the description in [63]. The different alignment kernels considered for the study are discrete time warping kernel (DTWK), dynamic time alignment kernel (DTAK), global alignment kernel (GAK) and triangular alignment kernel (TGAK). We have implemented the DTWK and DTAK as per the description in their respective papers. We have used the code given in the web page of the authors of GAK and TGAK for computing GAK and TGAK [115]. The DTWK, DTAK, GAK and TGAK are computed for different values of parameter δ in (13.77) for DTWK and δ in the Gaussian kernel used for the other alignment kernels. The TGAK is computed for different values of the parameter r in (13.81). For construction of HMM-based IMKs (HIMK) such as subsequence matching based HIMK (SSHIMK) and complete sequence matching HIMK (CSHIMK), the MLHMMs with the specifications as mentioned earlier are used. The LIB-SVM [80] tool is used to build the dynamic kernel based SVM classifiers. In this study, the one-against-the-rest approach is considered for 9-class E-set recognition task and 103-class CV segment recognition task using SVMs with FK and alignment kernels. The one-against-one approach is used for

9-class E-set recognition task using the LRK-based SVM, as LRK is computed for every pair of classes. The LRK-based SVM is not considered for the CV segment recognition task, because it leads to 10712 pairs of classes for 103 CV classes. Generating such a large number of LRK is computationally intensive. The value of trade-off parameter, \mathfrak{C} used to build the SVM classifiers is chosen empirically as 10.

The classification accuracies obtained for the MLHMM-based system, LMHMM-based system and SVM-based systems using score space kernels, alignment kernels and HMM-based IMKs for both E-set and CV segment recognition tasks are given in Table 13.8. The accuracies presented in Table 13.8 are the best accuracies observed among the CDHMM-based systems, the score space kernel based SVMs and the alignment kernel based SVMs for different values of N , Q , δ and r . It is seen that the LMHMM-based system gives a better performance than the MLHMM-based system. It is also seen that performance of the SVMs using score space kernels and HMM-based IMKs for E-set recognition is better than that of the MLHMM-based systems. This is mainly because the MLHMM-based classifier is trained using a non-discriminative learning based technique, where as the SVM classifiers using score space kernels and HMM-based IMKs are built using a discriminative learning based technique. However, the performance of SVMs using alignment kernels is observed to be poorer than that of MLHMM-based systems and SVMs using score space kernels and HMM-based IMKs for E-set recognition. It is seen that the performance of SVMs using score space kernels, alignment kernels and HMM-based IMKs

Table 13.8. Comparison of classification accuracy (CA) (in %), of the CDHMM-based classifiers and SVM-based classifiers using FK, LRK, DTWK, DTAK, GAK and TGAK for E-set and CV segment recognition tasks. CA95%CI indicates average classification accuracy along with 95% confidence interval. The pair (N,Q) indicates the values of number of states in HMM (N) and the number of components in each of the state-specific GMMs (Q). The pair (δ,r) indicates the values of the parameters δ and r used in alignment kernels [61]. NSV indicates the average number of support vectors.

Classification model	E-set recognition			CV segment recognition		
	$(N,Q)/(\delta,r)$	CA	NSV	$(N,Q)/(\delta,r)$	CA95%CI	NSV
MLHMM	(12,2)	90.30	-	(5,3)	48.87±0.17	-
LMHMM	(11,3)	92.22	-	(5,3)	50.13±0.13	-
SVM using	FK	(12,2)	91.57	265	(5,3)	52.58±0.16
	LRK	(12,2)	95.00	-	-	-
	DTWK	(1000,-)	85.37	626	(1000,-)	52.51±0.19
	DTAK	(15,-)	87.11	648	(15,-)	54.10±0.16
	GAK	(20,-)	87.41	639	(20,-)	54.76±0.15
	TGAK	(45,25)	87.30	635	(45,25)	54.77±0.17
	SSHIMK	(12,2)	92.83	631	(5,3)	55.02±0.95
	CSHIMK	(12,2)	93.33	591	(5,3)	55.36±0.85
						248

is better than MLHMM-based system for the recognition of CV segments. It is also seen that the performance of LRK-based SVMs is better than that of the SVM classifiers using FK, alignment kernels and HMM-based IMKs for E-set recognition. For the CV segment recognition, the SVM using CSHIMK performs better than the SVM using score space kernels and alignment kernels and marginally better than the SVM using SSHIMK.

13.9. Summary and conclusions

In this Chapter, we addressed the issues in designing dynamic kernels for analysis of varying length patterns extracted from speech or image data. A review of different approaches to design dynamic kernels for varying length patterns represented as sets or sequences of local feature vectors has been presented. The approaches to design of intermediate matching kernel (IMK) based dynamic kernels have been presented. The comparative studies on classification of varying length patterns represented as sets of feature vectors extracted from speech data have been carried out for speech emotion recognition and speaker identification tasks. The comparative studies on classification and matching of varying length patterns represented as sets of feature vectors extracted from image data have been carried out for image classification, image annotation, image matching and content based image retrieval tasks. Results of these studies demonstrate the effectiveness of the dynamic kernels based approaches in comparison to the Gaussian mixture model based approaches and static pattern classification or matching based approaches. The comparative studies on classification of varying length patterns represented as sequences of local feature vectors extracted from speech data have been carried out for E-set recognition and for recognition of consonant-vowel segments of speech. The effectiveness of the dynamic kernels based approaches in comparison to the hidden Markov model based approaches for these tasks has been demonstrated. The studies presented in this Chapter demonstrate the suitability of dynamic kernels based approaches to analysis of varying length patterns extracted from speech and image data.

References

- [1] C. J. C. Burges, A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*. **2**(2), 121–167 (June, 1998).
- [2] V. Wan and S. Renals. Evaluation of kernel methods for speaker verifica-

- tion and identification. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2002)*, pp. 669–672, Orlando, Florida, USA (May, 2002).
- [3] T. Jaakkola, M. Diekhans, and D. Haussler, A discriminative framework for detecting remote protein homologies, *Journal of Computational Biology.* **7** (1-2), 95–114 (February-April, 2000).
 - [4] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, Support vector machines using GMM supervectors for speaker verification, *IEEE Signal Processing Letters.* **13**(5), 308–311 (April, 2006).
 - [5] S. Bougħorbel, J. P. Tarel, and N. Boujemaa. The intermediate matching kernel for image local features. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2005)*, pp. 889–894, Montreal, Canada (July, 2005).
 - [6] S. Davis and P. Mermelstein, Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences, *IEEE Transactions on Acoustics, Speech and Signal Processing.* **28**(4), 357–366 (August, 1980).
 - [7] X. Huang, A. Acero, and H.-W. Hon, *Spoken Language Processing: a Guide to Theory, Algorithm, and System Development*. Prentice Hall, New Jersey (2001).
 - [8] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*. Pearson Education (2003).
 - [9] H. Hermansky, Perceptual linear predictive (PLP) analysis of speech, *Journal of the Acoustical Society of America.* **57**(4), 1738–1752 (April, 1990).
 - [10] B. Yegnanarayana, D. Saikia, and T. Krishnan, Significance of group delay functions in signal reconstruction from spectral magnitude or phase, *IEEE Transactions on Acoustics, Speech and Signal Processing.* **32**(3), 610–623 (January, 1984).
 - [11] H. A. Murthy and B. Yegnanarayana, Group delay functions and its application to speech technology, *Sadhana, Indian Academy of Sciences.* **36**(5), 745–782 (November, 2011).
 - [12] J. M. Henderson, Introduction to real-world scene perception, *Visual Cognition.* **12**(3), 849–851 (2005).
 - [13] C.-F. Tsai, Image mining with spectral features: A case study of scenery image classification, *Expert Systems with Applications.* **32**(1), 135–142 (January, 2007).
 - [14] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz, Efficient and effective querying by image content, *Journal of Intelligent Information Systems - Special Issue: Advances in Visual Information Management Systems.* **3**(3/4), 231–262 (July, 1994).
 - [15] K. Wu and K.-H. Yap, Fuzzy SVM for content-based image retrieval, *IEEE Computational Intelligence Magazine.* **1**(2), 10–16 (May, 2006).
 - [16] F. Jing, M. Li, H.-J.Zhang, and B. Zhang, A unified framework for image retrieval using keyword and visual features, *IEEE Transactions on Image Processing.* **14**(7), 979–989 (July, 2005).
 - [17] A. Oliva and A. Torralba, Building the gist of a scene: The role of global

- image features in recognition, *Progress in Brain research: Visual Perception*. **155**, 23–36 (2006).
- [18] K. Mele, D. Suc, and J. Maver, Local probabilistic descriptors for image categorisation, *IET Computer Vision*. **3**(1), 8–23 (March, 2009).
 - [19] R. Yan and A. G. Hauptmann, A review of text and image retrieval approaches for broadcast news video, *Information Retrieval*. **10**(4-5), 445–484 (October, 2007).
 - [20] Y. Rui, T. S. Huang, and S.-F. Chang, Image retrieval past, present and future, *Journal of Visual Communication and Image Representation*. **10**(1), 39–62 (March, 1999).
 - [21] J. Wang, J. Li, and G. Wiederhold, SIMPLIcity: Semantics-sensitive integrated matching for picture libraries, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **23**(9), 947–963 (September, 2001).
 - [22] G. Aggarwal, T. V. Ashwin, and S. Ghosal, An image retrieval system with automatic query modification, *IEEE Transactions on Multimedia*. **4**(2), 201–214 (June, 2002).
 - [23] C. Carson, S. Belongie, H. Greenspan, and J. Malik, Blobworld: Image segmentation using expectation maximization and its application to image querying, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **24**(8), 1026–1038 (August, 2002).
 - [24] J.-W. Hsieh and W. E. L. Grimson, Spatial template extraction for image retrieval by region matching, *IEEE Transactions on Image Processing*. **12**(11), 1404–1415 (November, 2003).
 - [25] F. Jing, M. Li, H.-J. Zhang, and B. Zhang, An efficient and effective region-based image retrieval framework, *IEEE Transaction on Image Processing*. **13**(5), 699–701 (May, 2004).
 - [26] J. Shi and J. Malik, Normalized cuts and image segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **22**(8), 888–905 (August, 2000).
 - [27] D. Comanicu and P. Meer, Mean shift: A robust approach toward feature space analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **24**(5), 603–619 (May, 2002).
 - [28] N. Ikonomakis, K. Plataniotis, M. Zervakis, and A. Venetsanopoulos. Region growing and region merging image segmentation. In *Proceedings of 13th International Conference on Digital Signal Processing*, vol. 1, pp. 299–302, Santorini, Greece (July, 1997).
 - [29] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings of 8th International Conference on Computer Vision*, vol. 2, pp. 416–423 (July, 2001).
 - [30] Y. Chen, J. Z. Wang, and R. Krovetz, CLUE: Cluster-based retrieval of images by unsupervised learning, *IEEE Transactions on Image Processing*. **14**(8), 1187–1201 (August, 2005).
 - [31] Z. M. Lu and H. Burkhardt, Color image retrieval based on DCT-domain vector quantization index histograms, *Electronic Letters*. **41**(17), 29–30 (August, 2005).

- [32] T. Kadir and M. Brady, Saliency, scale and image description, *International Journal of Computer Vision.* **45**(2), 83–105 (November, 2001).
- [33] D.G.Lowe, Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision.* **60**, 91–110 (2004).
- [34] J. Vogel and B. Schiele, Semantic modeling of natural scenes for content-based image retrieval, *International Journal of Computer Vision.* **72**(2), 133–157 (April, 2007).
- [35] S. L. Feng, R. Manmatha, and V. Lavrenko. Multiple Bernoulli relevance models for image and video annotation. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004)*, vol. 2, pp. 1002–1009, Washington DC, USA (June-July, 2004).
- [36] D. A. Reynolds, Speaker identification and verification using Gaussian mixture speaker models, *Speech Communication.* **17**, 91–108 (August, 1995).
- [37] D. Neiberg, K. Elenius, and K. Laskowski. Emotion recognition in spontaneous speech using GMMs. In *Proceedings of INTERSPEECH*, pp. 809–812, Pittsburgh, Pennsylvania, USA (September, 2006).
- [38] C. J. C. Burges, A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery.* **2**(2), 121–167 (June, 1998).
- [39] P.-Y. Oudeyer, The production and recognition of emotions in speech: Features and algorithms, *International Journal of Human-Computer Studies.* **59**, 157–183 (July, 2003).
- [40] S. Chandrakala and C. Chandra Sekhar. Combination of generative models and SVM based classifier for speech emotion recognition. In *Proceedings of the International Joint Conference on Neural Networks (IJCNN 2009)*, pp. 497–502, Atlanta, Georgia, USA (June, 2009).
- [41] A. Ganapathiraju, J. Hamaker, and J. Picone. Hybrid SVM/HMM architectures for speech recognition. In *Proceedings of the Sixth International Conference on Spoken Language Processing (ICSLP 2000)*, pp. 504–507, Beijing, China (October, 2000).
- [42] V. Wan and S. Renals. Evaluation of kernel methods for speaker verification and identification. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 669–672, Orlando, Florida, US (May, 2002).
- [43] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines and other Kernel-based Learning Methods*. Cambridge University Press (2000).
- [44] C. Chandra Sekhar, K. Takeda, and F. Itakura. Recognition of subword units of speech using support vector machines. In *Recent Research Developments in Electronics and Communication*, pp. 101–136. Transworld Research Network, Trivandrum, Kerala, India (2003).
- [45] S. Haykin, *Neural Networks and Learning Machines*. Third Edition, PHI Learning Private Limited (2010).
- [46] L. Kaufman. Solving the quadratic programming problem arising in support vector classification. In eds. B. Scholkopf, C. Burges, and A. Smola, *Advances in Kernel Methods: Support Vector Learning*, pp. 147–167. MIT Press, Cambridge, MA, USA (1999).

- [47] T. M. Cover, Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition, *IEEE Transactions on Electronic Computers*. **EC-14**(3), 326–334 (June, 1965).
- [48] B. Scholkopf, S. Mika, C. Burges, P. Knirsch, K.-R. Muller, G. Ratsch, and A. Smola, Input space versus feature space in kernel-based methods, *IEEE Transactions on Neural Networks*. **10**(5), 1000–1017 (September, 1999).
- [49] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, U.K. (2004).
- [50] K. M. Borgwardt. Graph kernels. Ph.D thesis, Faculty of Mathematics, Computer Science and Statistics, LudwigMaximilians Universität, Munich (July, 2007).
- [51] C. Leslie, E. Eskin, and W. S. Noble. The spectrum kernel: A string kernel for SVM protein classification. In *Proceedings of The Pacific Symposium on Biocomputing*, pp. 564–575, Lihue, Hawaii (January, 2002).
- [52] C. Leslie, E. Eskin, J. Weston, and W. S. Noble. Mismatch string kernels for SVM protein classification. In eds. S. Becker, S. Thrun, and K. Obermayer, *Proceedings of Advances in Neural Information Processing (NIPS 2002)*, pp. 1417–1424. MIT Press:Cambridge, MA, Vancouver, British Columbia, Canada (December, 2002).
- [53] C. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, Mismatch string kernels for discriminative protein classification, *Bioinformatics*. **20**(4), 467–476 (March, 2004).
- [54] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Christianini, and C. Watkins, Text classification using string kernels, *Journal of Machine Learning Research*. **2**, 419–444 (February, 2002).
- [55] K. Tsuda, T. Kin, and K. Asai, Mariginalized kernels for biological sequences, *Bioinformatics*. **18(suppliment 1)**, S268–S275 (2002).
- [56] E. L. Allwein, R. E. Schapire, and Y. Singer, Reducing multiclass to binary: A unifying approach for margin classifiers, *Journal of Machine Learning Research*. **1**, 113–141 (December, 2000). ISSN 1532-4435.
- [57] U. H.-G. Kressel. Pairwise classification and support vector machines (1999).
- [58] K.-A. Lee, C.H. You, H. Li, and T. Kinnunen. A GMM-based probabilistic sequence kernel for speaker verification. In *Proceedings of INTERSPEECH*, pp. 294–297, Antwerp, Belgium (August, 2007).
- [59] C. H. You, K. A. Lee, and H. Li, An SVM kernel with GMM-supervector based on the Bhattacharyya distance for speaker recognition, *IEEE Signal Processing Letters*. **16**(1), 49–52 (January, 2009).
- [60] H. Shimodaira, K.-I. Noma, M. Nakai, and S. Sagayama. Dynamic time-alignment kernel in support vector machine. In *Proceedings of Advances in Neural Information Processing Systems (NIPS 2002)*, pp. 921–928, MIT Press, Vancouver, BC, Canada (December, 2002).
- [61] M. Cuturi. Fast global alignment kernels. In eds. L. Getoor and T. Scheffer, *Proceedings of the 28th International Conference on Machine Learning (ICML 2011)*, pp. 929–936, ACM, New York, NY, USA (June, 2011).
- [62] N. Smith, M. Gales, and M. Niranjan. Data-dependent kernels in SVM clas-

- sification of speech patterns. Technical Report CUED/F-INFENG/TR.387, Cambridge University Engineering Department, Trumpington Street, Cambridge, CB2 1PZ, U.K. (April, 2001).
- [63] N. Smith and M. Gales. Speech recognition using SVMs. In *Proceedings of Advances in Neural Information Processing Systems (NIPS 2001)*, pp. 1197–1204, MIT Press, Vancouver, BC, Canada (December, 2001).
 - [64] T. Jebara, R. Kondor, and A. Howard, Probability product kernels, *Journal of Machine Learning Research.* **5**, 819–844 (December, 2004).
 - [65] S. Bougħorbel, J.-P. Tarel, and F. Fleuret. Non-Mercer kernels for SVM object recognition. In *Proceedings of British Machine Vision Conference (BMVC 2004)*, pp. 137–146, Kingston, UK (September, 2004).
 - [66] C. Wallraven, B. Caputo, and A. Graf. Recognition with local features: The kernel recipe. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV 2003)*, pp. 257–264, Nice, France (October, 2003).
 - [67] B. Haasdonk and C. Bahlmann. Learning with distance substitution kernels. In *Proceedings of DAGM'04: 26th Annual Pattern Recognition Symposium*, pp. 220–227, Tübingen, Germany (August-September, 2004).
 - [68] M. Cuturi, J. P. Vert, O. Birkenes, and T. Matsui. A kernel for time series based on global alignments. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2007)*, vol. 2, pp. 413–416, Honolulu, Hawaii, USA (April, 2007).
 - [69] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, Speaker verification using adapted Gaussian mixture models, *Digital Signal Processing.* **10**(1-3), 19–41 (January, 2000).
 - [70] A. D. Dileep and C. Chandra Sekhar, GMM-based intermediate matching kernel for classification of varying length patterns of long duration speech using support vector machines, *IEEE Transactions on Neural Networks and Learning Systems.* **25**(8), 1421–1432 (Aug, 2014).
 - [71] S. Kullback and R. A. Leibler, On information and sufficiency, *The Annals of Mathematical Statistics.* **22**(1), 79–86 (1951).
 - [72] M. G. Genton, Classes of kernels for machine learning: A statistics perspective, *Journal of Machine Learning Research.* **2**, 299–312 (December, 2001).
 - [73] A. Dileep and C. Sekhar, Hmm based intermediate matching kernel for classification of sequential patterns of speech using support vector machines, *IEEE Transactions on Audio, Speech, and Language Processing.* **21**(12), 2570–2582 (Dec, 2013).
 - [74] F. Burkhardt, A. Paeschke, M. Rolfs, and W. S. B. Weiss. A database of German emotional speech. In *Proceedings of INTERSPEECH*, pp. 1517–1520, Lisbon, Portugal (September, 2005).
 - [75] S. Steidl. Automatic classification of emotion-related user states in spontaneous children's speech. PhD thesis, Der Technischen Fakultät der Universität Erlangen-Nürnberg, Germany (2009).
 - [76] The NIST year 2002 speaker recognition evaluation plan, <http://www.itl.nist.gov/iad/mig/tests/spk/2002/> (2002).
 - [77] The NIST year 2003 speaker recognition evaluation plan,

- <http://www.itl.nist.gov/iad/mig/tests/sre/2003/> (2003).
- [78] N. Sato and Y. Obuchi, Emotion recognition using Mel-frequency cepstral coefficients, *Journal of Natural Language Processing*. **14**(4), 83–96 (2007).
 - [79] F. Sha and L. Saul. Large margin Gaussian mixture modeling for phonetic classification and recognition. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 265–268, Toulouse, France (May, 2006).
 - [80] C.-C. Chang and C.-J. Lin, LIBSVM: A library for support vector machines, *ACM Transactions on Intelligent Systems and Technology*. **2**(3), 27:1–27:27 (April, 2011). Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
 - [81] J. Vogel and B. Schiele. Natural scene retrieval based on a semantic modeling step. In *Proceedings of International conference on image and video retrieval (LNCS vol 3115)*, pp. 207–215, Dublin, Ireland (2004).
 - [82] A. Oliva and A. Torralba, Modeling the shape of the scene: A holistic representation of the spatial envelope, *International Journal of Computer Vision*. **42**(3), 145–175 (May, 2001).
 - [83] P. Duygulu, K. Barnard, J. F. G. de Freitas, and D. A. Forsyth. Object recognition as machine translation: Learning a lexicon for a fixed image vocabulary. In *Proceedings of European Conference on Computer Vision (ECCV 2002)*, vol. 4, pp. 97–112, Copenhagen, Denmark (May-June, 2002).
 - [84] J. Z. Wang, G. Wiedeshold, O. Firchein, and S. X. Wei, Content-based image indexing and searching using Daubechies's wavelets, *International Journal on Digital Libraries*. **1**(4), 311–328 (1998).
 - [85] G.-H. Liu, Z.-Y. Li, L. Zhang, and Y. Xu, Image retrieval based on micro-structure descriptor, *Pattern Recognition*. **44**(9), 2123–2133 (September, 2011).
 - [86] C. Schmid and R. Mohr, Local gray value invariants for image retrieval, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **19**(5), 530–535 (May, 1997).
 - [87] N. Rasiwasia, P. J. Moreno, and N. Vasconcelos, Bridging the gap: Query by semantic example, *IEEE Transactions on Multimedia*. **9**(5), 923–938 (August, 2007).
 - [88] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Wisley, New York (1991).
 - [89] F. Jing, M. Li, H.-J. Zhang, and B. Zhang, An efficient and effective region-based image retrieval framework, *IEEE Transactions on Image Processing*. **13**(5), 699–709 (May, 2004).
 - [90] F. Jing, M. Li, H. Zhang, and B. Zhang, Relevance feedback in region-based image retrieval, *IEEE Transaction on Circuits and Systems for Video Technology*. **14**(5), 672–681 (May, 2004).
 - [91] A. Marakakis, N. Galatsanos, A. Likas, and A. Stafylopatis, Probabilistic relevance feedback approach for content-based image retrieval based on Gaussian mixture models, *IET Image Processing*. **3**(1), 10–25 (February, 2009).

- [92] F. Odone and A. Verri. Image classification and retrieval with kernel methods. In eds. G. Camps-valls, J. L. Rojo-Alvarez, and M. Martinez-ramon, *Kernel Methods in Bioengineering, Signal and Image Processing*. Idea Group Publishing (2007).
- [93] M. J. Swain and D. H. Ballard, Color indexing, *International Journal of Computer Vision*. **7**(1), 11–32 (November, 1991).
- [94] G. Salton and C. Buckley, Term-weighting approaches in automatic text retrieval, *Information Processing and Management*. **24**(5), 513–523 (1988).
- [95] A. Martins. String kernels and similarity measures for information retrieval. Technical report, Priberam, Lisbon, Portugal (2006).
- [96] W. M. Campbell, J. P. Campbell, T. G. Gleason, D. A. Reynolds, and W. Shen, Speaker verification using support vector machines and high-level features, *IEEE Transactions on Audio, Speech, and Language Processing*. **15**(7), 2085 –2094 (September, 2007).
- [97] V. Wan and S. Renals. Evaluation of kernel methods for speaker verification and identification. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2002)*, pp. 669–672, Orlando, Florida, USA (May, 2002).
- [98] A. D. Dileep and C. C. Sekhar. Speaker recognition using intermediate matching kernel based support vector machines. In eds. A. Neustein and H. Patil, *Speaker Forensics: New Developments in Voice Technology to Combat and Detect Threats to Homeland Security*. Springer (2011).
- [99] C. D. Manning, P. Raghavan, and H. Schutze, *An Introduction to Information Retrieval*. Cambridge University Press, Cambridge, England (2008).
- [100] G. Schaefer and M. Stich. UCID- An uncompressed colour image database. In *Proceedings of SPIE 5307, Storage and Retrieval Methods and Applications for Multimedia*, vol. 472, San Jose, CA (December, 2004).
- [101] T. Deselaers, D. Keysers, and H. Ney, Features for image retrieval: An experimental comparison, *Information Retrieval*. **11**(2), 77–107 (April, 2008).
- [102] A. Makadia, V. Pavlovic, and S. Kumar, Baselines for image annotation, *International Journal of Computer Vision*. **90**(1), 88–105 (October, 2010).
- [103] M. Guillaumin, T. Mensink, J. Verbeek, and C. Schmid. TagProp: Discriminative metric learning in nearest neighbor models for image auto-annotation. In *Proceedings of IEEE International Conference on Computer Vision (ICCV 2009)*, pp. 309–316, Kyoto, Japan (October, 2009).
- [104] H. Fu, Q. Zhang, and G. Qiu. Random forest for image annotation. In *Proceedings of European Conference on Computer Vision (ECCV 2012)*, vol. 6, pp. 86–99, Florence, Italy (October, 2012).
- [105] K. Barnard, P. Duygulu, and D. Forsyth. Recognition as translating images into text. In *Proceedings of Internet Imaging IX, Electronic Imaging 2003 (Invited paper)*, Santa Clara, California, USA (February, 2003).
- [106] G. Carneiro, A. B. Chan, P. J. Moreno, and N. Vasconcelos, Supervised learning of semantic classes for image annotation and retrieval, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **29**(3), 394–410 (March, 2007).
- [107] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, Learning multi-label

- scene classification, *Pattern Recognition*. **37**(9), 1757–1771 (September, 2004).
- [108] G. Tsoumakas, I. Katakis, and I. Vlahavas. Mining multi-label data. In eds. L. Rekach and O. Maimon, *Data Mining and Knowledge Discovery Handbook*, pp. 667–686. Springer (2010).
 - [109] D. M. J. Tax and R. P. W. Duin. Using two-class classifiers for multiclass classification. In *Proceedings of IEEE Computer Society International Conference on Pattern Recognition (ICPR 2002)*, vol. 2, pp. 124 – 127, Quebec, Canada, (August, 2002).
 - [110] URL <http://www.cs.washington.edu/research/imagedatabase/groundtruth/>.
 - [111] ISOLET Corpus. *Release 1.1*. Center for Spoken Language Understanding, Oregon Graduate Institute (July, 2000).
 - [112] S. V. Gangashetty. Neural network models for recognition of consonant-vowel units of speech in multiple languages. PhD thesis, Department of Computer Science and Engineering, IIT Madras, Chennai (February, 2005).
 - [113] P. C. Loizou and A. S. Spanias, High-performance alphabet recognition, *IEEE Transactions on Speech and Audio Processing*. **4**(6), 430–445 (1996).
 - [114] H. Jiang, X. Li, and C. Liu, Large margin hidden Markov models for speech recognition, *IEEE Transactions on Audio, Speech, and Language Processing*. **14**(5), 1584–1595 (September, 2006).
 - [115] M. Cuturi. Triangular global alignment kernels. URL <http://www.iip.ist.i.kyoto-u.ac.jp/member/cuturi/GA.html>.

Chapter 14

Fuzzy Rough Granular Neural Networks for Pattern Analysis

Avatharam Ganivada^a, Shubhra Sankar Ray^{a,b} and Sankar K. Pal^{a,b}

^a*Center for Soft Computing Research
Indian Statistical Institute, India*

^b*Machine Intelligence Unit
Indian Statistical Institute, India
sankar@isical.ac.in*

Granular computing is a computational paradigm in which a granule represents a structure of patterns evolved by performing operations on the individual patterns. Two granular neural networks are described for performing the pattern analysis tasks like classification and clustering. The granular neural networks are designed by integrating fuzzy sets and fuzzy rough sets with artificial neural networks in a soft computing paradigm. The fuzzy rough granular neural network (FRGNN) for classification is based on multi-layer neural networks. On the other hand, the fuzzy rough granular self-organizing map (FRGSOM) for clustering is based on self-organizing map. While the FRGNN uses the concepts of fuzzy rough set for defining its initial connection weights in supervised mode, the FRGSOM, as its name implies, exploits the same in unsupervised manner. Further, the input vector of FRGNN & FRGSOM and the target vector of FRGNN are determined using the concepts of fuzzy sets. The performance of FRGNN and FRGSOM is compared with some of the related methods using several life data sets.

14.1. Introduction

Natural Computing is a consortium of different methods and theories that have emerged from natural phenomena such as brain modeling, self-organization, self-repetition, self-evaluation, self-reproduction, group behaviour, Darwinian survival, granulation and perception. Based on the tasks abstracted from these phenomena, various computing

paradigms/technologies like artificial neural networks, fuzzy logic, rough logic, evolutionary algorithms, fractal geometry, DNA computing, quantum computing and granular computing are developed. They take insanitation from the nature for their computation, and are used for solving real life problems. For example, granulation abstracted from natural phenomena possesses perception based information representation which is an inherent characteristic of human thinking and reasoning process. One can refer to [1] and [2] for using fuzzy information granulation as a way to natural computing.

Granular computing (GrC) provides an information processing framework [3–6] where computation and operations are performed on information granules. Information granules are formulated by abstracting the common properties of patterns in data such as similarity, proximity and equality. GrC is based on the realization that precision is sometimes expensive and not very meaningful in modeling and controlling complex systems [7]. When a problem involves incomplete, uncertain, and vague information, one may find it convenient to consider granules, instead of the distinct elements, for its handling. Accordingly, granular computing became an effective framework for the design and implementation of efficient and intelligent information processing systems for various real life decision-making applications. The said framework may be modeled in soft computing using fuzzy sets, rough sets, artificial neural networks and their integrations, among other theories.

Information granulation using fuzzy sets can be found in [8] and [9]. Rough set [10–12] computes rough information granules induced by crisp equivalence classes. The crisp equivalence class represents a clump of objects obtained by grouping them with equal values (indiscernibility relation) corresponding to a feature. These are used to find lower and upper approximations which are exploited in approximating a set. As crisp equivalence classes are essential to rough sets, fuzzy equivalence granules, represented by fuzzy similarity relations, are important to fuzzy rough sets. In fuzzy rough set, the set is characterized by lower and upper approximations [13, 14]. The patterns in the set posses membership values belonging to the approximations computed using fuzzy equivalence granules and fuzzy decision classes. The fuzzy decision classes incorporate the average of memberships of patterns in each class. The memberships for belonging to the decision classes are computed by finding the weighted distances from all patterns in the class to its mean, and normalizing the distances with the help of fuzzyfiers. The use fuzzy information granulation is to efficiently

handle uncertainty arising in the overlapping classes and incompleteness in the class information.

Several researchers have incorporated the information granulation into artificial neural networks to develop different granular neural networks [15, 16]. Methods [17, 18] for classification are based on the integration of fuzzy information granulation and artificial neural networks. A fuzzy rule based neural network for linguistic data fusion is designed in [17]. In [18], an interval valued fuzzy membership function is used to design a granular network for classification where the network is trained using interval valued evolutionary algorithm. Different rule based neural networks for classification using rough sets are developed in [19–21]. The rules are generated using the concepts of rough sets. An interval set representation of clusters of web users of three educational web sites are evolved in [22] using self-organizing map, where the lower and upper approximations are computed at the nodes in the output layer of the self-organizing map. The granular hierarchical self-organizing map [23] uses a bidirectional update propagation method to modify its network parameters. A fuzzy self organizing map is designed in [24] as a fuzzy classifier using the concept of fuzzy sets. The fuzzy rough rule based neural networks for pattern recognition tasks are designed in [25]- [26]. The input of the networks [25, 27] is defined in terms of 3D input vector using fuzzy set. The network [26] uses the normalized data as its input. The dependency factors of the features (attributes) defined using fuzzy rough sets are encoded into the networks as the initial connection weights.

This article describes two different granular neural networks, developed using fuzzy sets and fuzzy rough sets, for classification and clustering tasks. The network in [25] is developed for classification task and is based on a granular input vector and network parameters. While the fuzzy set defines the input vector of the network in terms of 3D granular vector, the fuzzy rough set determines dependency factors of the features and these are used as the network parameters (initial connection weights). The network consists of input, hidden and output layers and is referred as FRGNN. The number of nodes in the input layer of the network is set equal to 3D features (granular input vector), corresponding to 3 components *low*, *medium* and *high*. The number of nodes in the hidden and output layers are set equal to the number of classes. The connection weights between the input & hidden layers and the hidden & output layers are initialized with the dependency factors of the 3D features. A gradient decent method is used for training the network. During training the weights are modified. The

weights generate a non linear decision boundary by which the overlapping patterns are efficiently classified.

The network is designed in [27] for identifying underlying cluster structures in the data and is based on a granular input vector, initial connection weights and the conventional self-organizing map (SOM). In the process of designing the network, fuzzy sets and fuzzy rough sets are employed. While fuzzy sets are used to develop linguistic input vectors representing fuzzy information granules, fuzzy rough sets are used to extract the crude domain knowledge about data in terms of dependency factors. The dependency factor for every feature using fuzzy rough set is defined in unsupervised manner. The initial connection weights are incorporated into the SOM. The initial knowledge based network having the granular inputs is called fuzzy rough granular self organizing map (FRGSOM) and is trained through the competitive learning process of the conventional SOM. After completion of the training process, the network determines the underlying granulation structures/clusters of the data. These are formed at the nodes of the output layer of the FRGSOM in a topological order. The initial connection weights of FRGSOM enable it to cope with uncertainty arising in overlapping regions between cluster boundaries.

A brief description of fuzzy rough granular neural network for classification is provided in Section 14.2. The formation of fuzzy rough granular self organizing map for clustering is discussed in Section 14.3. This is followed by experimental results section. The last section provides discussions and conclusions of this investigation.

14.2. Architecture of Fuzzy Rough Granular Neural Network for Classification

The fuzzy rough granular neural network (FRGNN) [25] for classification is designed by integrating the concepts of fuzzy sets, fuzzy rough sets and neural networks. While fuzzy sets are used for defining input vector and target vector of the network, the fuzzy rough sets are utilized for determining initial connection weights of the network. While the input vector is represented in terms of granular form, the target vector is expressed in terms of membership value and zeros corresponding to a pattern using fuzzy sets. The network is trained using gradient decent method of multi-layer perception.

14.2.1. Back Propagation Algorithm for Multi-layered Neural Network

For updating the initial connection weights of the fuzzy rough granular neural network, the back propagation algorithm of multi-layer perceptron is used. The algorithm is as follows:

Input:

D: Data set with training patterns in the granular form and their associated target vectors in terms of membership values and zeros (see Section 14.2.3.1 for details).

η : Learning rate

α : Momentum term

$b_l = b$: Bias term which is kept constant at each node (l) in the hidden and output layers

Method:

(1) Assign initial weights, in terms of dependency factors, between the nodes (units) in all the layers of the network;

(2) While the network is trained for all training patterns for a particular number of iterations {

Forward propagation:

(3) For each unit j of the input layer, the output, say x_j , of an input unit, say, I_j is

{

$x_j = I_j$;

}

(4) For each unit l of the hidden or output layers, compute the output o_l of each unit l with respect to the previous layer, j , as

{

$$o_l = \sum_j w_{lj} x_j + b;$$

}

(5) Apply logistic activation function to compute the output of l th node in the hidden or output layers

{

$$\phi(x_l) = \frac{1}{1+e^{-o_l}};$$

}

Back propagation:

- (6) For each unit in the output layer, say k , compute the error using

$$\left\{ \begin{array}{l} Error_k = \phi(x_k)(1 - \phi(x_k))(T_k - \phi(x_k)), \text{ where } T_k \text{ denotes the} \\ \text{target value for } k\text{th unit in the output layer.} \end{array} \right.$$
 - (7) Compute the error for l th node in the hidden layer by using the error obtained at k th node in the output layer

$$\left\{ \begin{array}{l} \gamma_l = \phi(x_l)(1 - \phi(x_l)) \sum_k Error_k w_{kl}; \end{array} \right.$$
 - (8) Update the weight w_{lj} in the network using

$$\left\{ \begin{array}{l} \Delta w_{lj} = (\eta)x_j \gamma_l; \\ \Delta w_{lj}(t) = (\eta)x_j \gamma_l + (\alpha)\Delta w_{lj}(t - 1); \end{array} \right.$$
 - (9) Update bias b in the network using

$$\left\{ \begin{array}{l} \Delta b = (\eta)\gamma_l; \\ b(t) += \Delta b(t - 1); \end{array} \right.$$
- } /* end while loop for step 2*/

Output:

Classified patterns

Here the momentum term α is used to escape the local minima in the weight space. The value of bias b , chosen within 0 to 1, is kept constant at each node of the hidden and output layers of the network and t denotes the number of iterations. For every iteration, the training data is presented to the network for updating the weight parameters w_{lj} and bias b . The resulting trained network is used for classifying the test patterns.

14.2.2. Input Vector Representation in terms of 3D Granular Vector

A pattern F in $A \subseteq U$ is assigned a grade of membership value, denoted by $\mu_A(F)$, to a fuzzy set A , using a membership function as

$$A = \{(\mu_A(F), F), F \in U, \mu_A(F) \in [0, 1]\}, \forall F \in R^n. \quad (14.1)$$

The π membership function is defined as [28]

$$\pi(F, C, \lambda) = \begin{cases} 2(1 - \frac{\|F - C\|_2}{\lambda})^2, & \text{for } \frac{\lambda}{2} \leq \|F - C\|_2 \leq \lambda, \\ 1 - 2(\frac{\|F - C\|_2}{\lambda})^2, & \text{for } 0 \leq \|F - C\|_2 \leq \frac{\lambda}{2}, \\ 0, & \text{otherwise,} \end{cases} \quad (14.2)$$

where $\lambda > 0$ is a scaling factor (radius) of the π function with C as a central point and $\|\cdot\|_2$ denotes the Euclidian norm. An n -dimensional i th pattern F_i is represented by $3n$ -dimensional linguistic vector [16] as

$$\vec{F}_i = [\mu_{low(F_{i1})}(\vec{F}_i), \mu_{medium(F_{i1})}(\vec{F}_i), \mu_{high(F_{i1})}(\vec{F}_i), \dots, \mu_{high(F_{in})}(\vec{F}_i)], \quad (14.3)$$

where μ_{low} , μ_{medium} and μ_{high} indicate the membership values corresponding to the linguistic terms *low*, *medium* or *high* along each feature axis. The selection for the values of C and λ of π functions is explained in the following section.

14.2.2.1. Choice of Parameters of π Functions

Let $\{F_{ij}\}$ for $i = 1, 2, \dots, s$; $j = 1, 2, \dots, n$; denote a set of s patterns for data with n features, and $F_{j_{min,m}}$ and $F_{j_{max,m}}$ denote the minimum and maximum values along the j th feature considering all the s patterns. The average of all the s patterns along the j th feature, F_j , is initially computed in a way similar to [25] and is considered as the center of the linguistic term *medium* along that feature, denoted by r_{m_j} , as in Fig. 14.1. Then, the average values (along the j th feature F_j) of the patterns in the ranges $[F_{j_{min,m}}, r_{m_j}]$ and $(r_{m_j}, F_{j_{max,m}}]$, corresponding to the feature F_j , are defined as the means of the linguistic terms *low* and *high*, and denoted

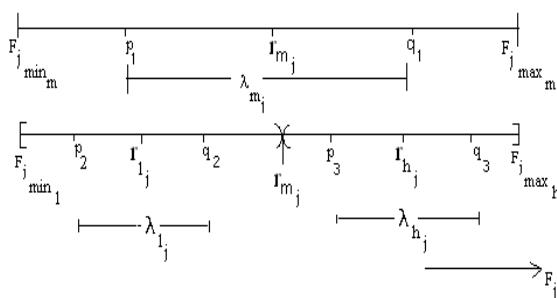


Fig. 14.1. Parameters of linguistic terms, *low*, *medium* and *high*.

by r_{l_j} and r_{h_j} , respectively. Again, using the same ranges we find the minimum and maximum for each of the ranges. While for the first range (*low*) the minimum and maximum are $F_{j_{min_m}}$ and r_{m_j} , respectively, for the second range (*high*) these are r_{m_j} and $F_{j_{max_m}}$. Then the center C and the corresponding scaling factor λ for linguistic terms *low*, *medium* and *high* along the j th feature F_j are defined as follows [25].

$$\begin{aligned} C_{medium_j} &= r_{m_j}, \\ p_1 &= C_{medium_j} - \frac{F_{j_{max_m}} - F_{j_{min_m}}}{2}, \\ q_1 &= C_{medium_j} + \frac{F_{j_{max_m}} - F_{j_{min_m}}}{2}, \\ \lambda_{m_j} &= q_1 - p_1, \\ \lambda_{medium} &= \frac{\sum_{j=1}^n \lambda_{m_j}}{n}. \end{aligned} \quad (14.4)$$

$$\begin{aligned} C_{low_j} &= r_{l_j}, \\ p_2 &= C_{low_j} - \frac{r_{m_j} - F_{j_{min_m}}}{2}, \\ q_2 &= C_{low_j} + \frac{r_{m_j} - F_{j_{min_m}}}{2}, \\ \lambda_{l_j} &= q_2 - p_2, \\ \lambda_{low} &= \frac{\sum_{j=1}^n \lambda_{l_j}}{n}. \end{aligned} \quad (14.5)$$

$$\begin{aligned} C_{high_j} &= r_{h_j}, \\ p_3 &= C_{high_j} - \frac{F_{j_{max_m}} - r_{m_j}}{2}, \\ q_3 &= C_{high_j} + \frac{F_{j_{max_m}} - r_{m_j}}{2}, \\ \lambda_{h_j} &= q_3 - p_3, \\ \lambda_{high} &= \frac{\sum_{j=1}^n \lambda_{h_j}}{n}. \end{aligned} \quad (14.6)$$

Figure 14.1 depicts the parameters, center and corresponding scaling factor, for linguistic terms *low*, *medium* or *high* along the feature F_j . Figure 14.2 shows a pattern \vec{F}_i having membership values to the linguistic properties *low*, *medium* or *high*. For one of these properties, the pattern would have strong membership value.

14.2.3. Class Memberships as Output Vectors

The membership of the i th pattern to k th class is calculated in [28] using

$$\mu_k(\vec{F}_i) = \frac{1}{1 + (\frac{Z_{ik}}{f_d})^{f_e}}, \quad (14.7)$$

where Z_{ik} is the weighted distance, and f_d & f_e are the denominational & exponential fuzzy generators, respectively, controlling membership of the

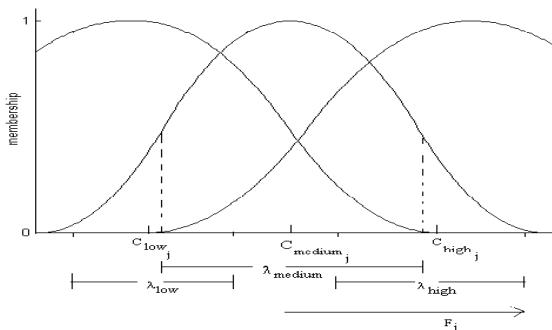


Fig. 14.2. Overlapping linguistic π sets.

patterns in the class. The weighted distance Z_{ik} is computed [16] using

$$Z_{ik} = \sqrt{\sum_{j=1}^n \left[\sum_{g=1}^3 \frac{1}{3} (\mu_g(F_{ij}) - \mu_g(O_{kj}))^2 \right]}, \quad \text{for } k = 1, 2, \dots, c, \quad (14.8)$$

where $\mu_g(F_{ij})$ represents the membership values of j th feature, corresponding to the linguistic terms *low*, *medium* and *high*, of i th pattern, O_{kj} is the mean of the k th class, and c is the number of classes.

14.2.3.1. Applying Membership Value to Target Vector

The target vector corresponding to i th pattern in a class contains one non zero membership value for one output node, representing that class, and zeros for the remaining output nodes, representing the remaining classes. That is, for the i th training pattern from the k th class, the target vector for k th output node is defined as [25]

$$d_k = \begin{cases} \mu_{INT}(\vec{F}_i), & \text{if } i\text{th pattern is from } k\text{th class denoting } k\text{th output node,} \\ 0, & \text{otherwise.} \end{cases} \quad (14.9)$$

14.2.4. Determining Initial Connection Weights of Network using Fuzzy Rough Sets

In fuzzy rough set theory, a similarity between two objects in U is modeled by a fuzzy tolerance relation R , that is,

$$\begin{aligned} R(x, x) &= 1 \quad (\text{reflexive}), \\ R(x, y) &= R(y, x) \quad (\text{symmetry}), \text{ and} \\ T(R(x, y)R(y, z)) &\leq R(x, z) \quad (T\text{-transitivity}), \end{aligned}$$

for all x, y and z in U . The fuzzy similarity relations are commonly considered to measure the approximate equality of objects. Given a t -norm, the relation R is called a fuzzy reflexive relation when the relation R does satisfy the properties of symmetry and T -transitivity.

14.2.4.1. Defining Fuzzy Reflexive Relations Corresponding to Features

Let $S = (U, \mathcal{A} \cup \{D\})$ denote a decision system. The 3D features, say $\{f_1, f_2, \dots, f_{3n}\}$, are represented by \mathcal{A} . If there are c classes then the belongingness of a pattern to each of the classes can be represented by $\{D\} = \{X_k, k = 1, 2, \dots, c\}$ where X_k is a decision feature containing membership values for all the patterns in class k . For every feature f , the fuzzy reflexive relation R between any two objects x and y is defined as [27]

$$R_f(x, y) = \begin{cases} \max \left(\min \left(\frac{f(y)-f(x)+\sigma_{f_{k_1}}}{\sigma_{f_{k_1}}}, \frac{f(x)-f(y)+\sigma_{f_{k_1}}}{\sigma_{f_{k_1}}} \right), 0 \right), \\ \quad \text{if } f(x) \text{ and } f(y) \in R_D(X_{k_1}) \\ \quad (\text{i.e., both patterns } x \text{ and } y \text{ belong to one particular decision class}), \\ \max \left(\min \left(\frac{f(y)-f(x)+\sigma_{f_{k_2}}}{\sigma_{f_{k_2}}}, \frac{f(x)-f(y)+\sigma_{f_{k_2}}}{\sigma_{f_{k_2}}} \right), 0 \right), \\ \quad \text{if } f(x) \in R_D(X_{k_1}), f(y) \in R_D(X_{k_2}) \\ \quad (\text{i.e., both patterns } x \text{ and } y \text{ belong to two different decision classes}) \\ \quad \text{and } k_1 \neq k_2, \end{cases} \quad (14.10)$$

where $\sigma_{f_{k_1}}$ & $\sigma_{f_{k_2}}$ represent the standard deviation of the patterns in the classes k_1 and k_2 , corresponding to feature f with respect to the decision classes $R_D(X_{k_1})$ and $R_D(X_{k_2})$, respectively, k_1 & $k_2 = 1, 2, \dots, c$.

14.2.4.2. Defining Fuzzy Decision Classes and Rough Representation

Suppose the i th pattern \vec{F}_i (see Section 14.2.2) is represented with \vec{x}_i , and O_{kj} & V_{kj} , $j = 1, 2, \dots, 3n$, denote the mean and the standard deviation, respectively, of the patterns belonging to k th class. The weighted distance Z_{ik} of a pattern \vec{x}_i , $i = 1, 2, \dots, s$, from the mean of the k th class is

defined as [28]

$$Z_{ik} = \sqrt{\sum_{j=1}^{3n} \left[\frac{x_{ij} - O_{kj}}{V_{kj}} \right]^2}, \text{ for } k = 1, 2, \dots, c, \quad (14.11)$$

where x_{ij} is the j th feature (3D feature) of i th pattern. Here s is the total number of patterns. The membership value of the i th pattern in the k th class, $\mu_k(\vec{x}_i)$, is computed using Eq. 14.7. The decision features representing fuzzy decision classes are defined as follows [25]:

i) Compute the average of the membership values of all the patterns in the k th class to its own class and replace the individual membership with that average. The average membership of patterns in the k th decision class is defined as

$$D_{kk} = \frac{\sum_{i=1}^{m_k} \mu_k(\vec{x}_i)}{|m_k|}, \text{ if } k = l, \quad (14.12)$$

where $|m_k|$ indicates the number of patterns in the k th decision class, and k and $l = 1, 2, \dots, c$. ii) Calculate the average of the membership values of all the patterns in the k th class to the other classes and replace the membership values with the corresponding average values. The average membership values for the patterns from their own class to other classes are computed as

$$D_{kl} = \frac{\sum_{i=1}^{m_k} \mu_l(\vec{x}_i)}{|m_k|}, \text{ if } k \neq l, \quad (14.13)$$

The fuzzy decision classes are defined as [25]

$$R_f(x, y) = \begin{cases} D_{kk}, & \text{if } f(x) = f(y), \\ D_{kl}, & \text{otherwise,} \end{cases} \quad (14.14)$$

for all x and y in U . Here D_{kk} corresponds to an average membership value of all the patterns that belong to the same class ($k = l$), and D_{kl} corresponds to the average membership values of all the patterns from classes other than k ($k \neq l$). The lower and upper approximations of a set $A \subseteq U$ are defined as [29]

$$(R_f \downarrow R_D)(y) = \inf_{x \in U} I(R_f(x, y), R_D(x)) \text{ and} \quad (14.15)$$

$$(R_f \uparrow R_D)(y) = \sup_{x \in U} T(R_f(x, y), R_D(x)), \quad (14.16)$$

for all y in U . For $B \subseteq \mathcal{A}$, the fuzzy positive region based on fuzzy B -indiscernibility relation is defined as

$$POS_B(y) = (R_B \downarrow R_D)(y), \quad (14.17)$$

for all $y \in U$. The dependency value for every feature in the set $B \subseteq \mathcal{A}$, denoted by γ_B , is defined as

$$\gamma_B = \frac{\sum_{x \in U} POS_B(x)}{|U|}, \quad (14.18)$$

where $|\cdot|$ denotes the cardinality of U , and $0 \leq \gamma_B \leq 1$.

14.2.4.3. Knowledge Encoding Procedure

We compute c decision tables $S_l = (U_l, \mathcal{A} \cup \{D\})$, $l = 1, 2, \dots, c$ using the training data. Each of the decision tables consists of patterns from each of the c classes where the patterns are added sequentially in the tables. Each pattern in S_l has $3n$ -dimensional features. The decision table S_l satisfies the following three conditions:

$$\text{i)} U_l \neq \emptyset, \quad \text{ii)} \bigcup_l^c U_l = U, \quad \text{and iii)} \bigcap_{l=1}^c U_l = \emptyset. \quad (14.19)$$

The number of patterns in U_l belongs to the c classes, say U_{de_k} , $k = 1, 2, \dots, c$. The size of the S_l , $l = 1, 2, \dots, c$ is dependent on the available number of objects from all the classes. Based on the c decision tables, the initial connection weights of the network are determined using fuzzy rough sets. We explain the procedure for determining the initial connection weights, based on the decision table S_1 when $l = 1$, as an example. The procedure is as follows.

S1: Find fuzzy reflexive relation matrix using Eq. 14.10 for every feature.

Here each row in the matrix represents fuzzy equivalence granule.

S2: Compute fuzzy decision classes using Eq. 14.14.

S3: Calculate the lower approximations of patterns in a set (class) using Eq. 14.15 for each feature, based on S1 and S2.

S4: Calculate the dependency value using Eq. 14.18 for each feature. Initialize the resulting values as initial connection weights between the input nodes and one hidden node as follows:

Let γ_i , $i = 1, 2, \dots, n$; denote the dependency degree of i th feature (a_i) in the decision table S_1 . The weight between one hidden node, representing one decision table (S_1), and i th input node is denoted by w_{1i} . Therefore, the connection weight between i th input node and the hidden node (w_{1i}), represented with $\gamma_i^{S_1}$, is defined as

$$\gamma_i^{S_1} = \frac{\sum_{x \in U} POS_i(x)}{|U|}. \quad (14.20)$$

S5: The connection weights between the hidden node and the nodes in the output layer are initialized as follows: Let $\beta_k^{S_1}$ denote the average of dependency factors of all the features with respect to the decision class k . The connection weight between the hidden node and k th output node is denoted by w_{k1} , $k = 1, 2, \dots, c$. Therefore, the weight w_{k1} is

initialized with $\beta_{S_1}^k$ as

$$\beta_{S_1}^k = \frac{\sum_{i=1}^n \gamma_i^1}{|n|}, \quad (14.21)$$

where γ_i^1 is defined as

$$\gamma_i^1 = \frac{\sum_{x \in U_{de_k}} POS_i(x)}{|U_{de_k}|}, \quad k = 1, 2, \dots, c. \quad (14.22)$$

Similarly, we define the dependency degrees, $\gamma_i^{S_l}$ and $\beta_{S_l}^k$ using the decision tables S_l , $l = 2, \dots, c$. The weights between i th input node & l th hidden node of the network and l th hidden node & k th output node are initialized with $\gamma_i^{S_l}$ and $\beta_{S_l}^k$, respectively. The connection weights of the network are refined using the gradient decent method (back propagation algorithm of MLP).

14.3. Fuzzy Rough Granular Self-Organizing Map

Here, the methodology of fuzzy rough granular self-organizing map (FRG-SOM) for clustering is described. The input vector of FRGSOM, in terms of 3D granular input vector corresponding to linguistic terms *low* *medium* and *high*, is defined using Eq. 14.3. The granular input data is used to generate granulation structures on basis of α -cut. The method involving α -cut for generating the granulation structures is developed as follows [27].

Method: Let x_{1j} and x_{2j} , $j = 1, 2, \dots, 3n$; denote two patterns with $3n$ dimensional features. The similarity value between the patterns, x_{1j} and x_{2j} , using fuzzy implication operator and t -norm, is calculated as

- S1) $I_{1j} \leftarrow \min(1 - x_{1j} + x_{2j}, 1)$, $I_{2j} \leftarrow \min(1 - x_{2j} + x_{1j}, 1)$. /* use fuzzy implication */
- S2) $T_j \leftarrow \max(I_{1j} + I_{2j} - 1, 0)$. /* use t -norm */
- S3) $m_{12} \leftarrow \min\{T_j\}$.

The similarity values for all possible pairs of patterns are computed and a similarity matrix of size $s \times s$ is constructed, where s is the total number of patterns in the data. An α -value, chosen within 0 to 1, is applied on the matrix and p number of granules are generated. The process of determining the granules by using α -value on the matrix is described in [27]. We arrange the p groups according to their size in decreasing order, where the size is the number of patterns in the group. The top c groups, out of the p groups, are selected. The patterns in $c + 1$ to p groups are added into the top c -groups

according to their closeness to the cluster centers of the top c groups. The resultant c groups are employed in defining the initial connection weights of FRGSOM.

14.3.1. Determining Initial Connection Weights of FRGSOM

The c -groups labeled with the numbers as its crisp decision classes are presented to a decision table $S = (U, \mathcal{A} \cup \{D\})$. Here, $\mathcal{A} = \{f_1, f_2, \dots, f_{3n}\}$ characterizes $3n$ dimensional features and $\{D\}$ represents decision classes. Based on the decision table S , the initial connection weights of FRGSOM using fuzzy rough sets are determined as follows [27].

- S1) Calculate fuzzy reflexive relational matrix using Eq. 14.10 for every feature f_j , $j = 1, 2, \dots, 3n$.
- S2) Compute fuzzy decision classes corresponding to crisp decision classes in two steps.
 - a) The membership values of all the patterns in the k th class to its own class is defined as

$$D_{kk} = \mu_k(\vec{x}_i), \text{ if } k = l, \quad (14.23)$$

where $\mu_k(\vec{x}_i)$ represents the membership value of the i th pattern to the k th class calculated using Eq. 14.7, and

b) the membership values of all patterns in the k th class to other classes is defined as

$$D_{kl} = 1, \text{ if } k \neq l, \quad (14.24)$$

where k and $l = 1, 2, \dots, c$. For any two patterns x and $y \in U$, with respect to a feature $f \in \{D\}$, the fuzzy decision classes are specified as

$$R_D(x, y) = \begin{cases} D_{kk}, & \text{if } f(x) = f(y), \\ D_{kl}, & \text{otherwise.} \end{cases} \quad (14.25)$$

- S3) Compute lower approximations of each set (granule) using Eq. 14.15 for every feature f_j , $j = 1, 2, \dots, 3n$.
- S4) Calculate the dependency value using Eq. 14.18 for every feature f_j , $j = 1, 2, \dots, 3n$; with respect to each decision class. Assign the resulting dependency factors as initial weights, say w_{kj} , $j = 1, 2, \dots, 3n$; $k = 1, 2, \dots, c$ between $3n$ nodes in the input layer and c nodes in output layer of FRGSOM. Here c represents the number of clusters.

14.3.2. Training of FRGSOM

The FRGSOM is trained using the competitive learning of the conventional SOM [30]. The procedure for training FRGSOM is as follows.

- S1) Present an input vector, $x_j(t)$, $j = 1, 2, \dots, 3n$; at the nodes in the input layer of FRGSOM for a particular number of iteration t .
- S2) At t th iteration, compute the Euclidian distance between the input vector, $x_j(t)$, and weight vector $w_{kj}(t)$ at the k th output node using

$$L_k = \|x_j(t) - w_{kj}(t)\|^2. \quad (14.26)$$

- S3) Find the winner neuron p using

$$p = \operatorname{argmin}\{L_k\}, k = 1, 2, \dots, c. \quad (14.27)$$

- S4) Find neighborhood neurons, N_p , around the winner neuron p using the Gaussian neighborhood function as

$$N_p(t) = \exp\left(-\frac{\Lambda_{pk}^2}{2\sigma(t)^2}\right), \quad (14.28)$$

where $\Lambda_{pk}(t)$ is the distance between the position of winning neuron p and the position of a neuron k , lying in the neighborhood of N_p , in two dimensional array. The width of the neighborhood set σ at iteration t is defined as

$$\sigma(t) = \sigma_0 \exp\left(-\left(\frac{t}{\tau_1}\right)\right). \quad (14.29)$$

Here the value of σ is decreased with every value of iteration t and τ_1 is a constant chosen as in [30].

- S5) Update the connection weights of the neighborhood neurons N_p using
- $$w_{kj}(t+1) = \begin{cases} w_{kj}(t) + \alpha(t)N_p(t)(x_j(t) - w_{kj}(t)), & \text{if } k \in N_p(t), \\ w_{kj}(t), & \text{else.} \end{cases} \quad (14.30)$$

The learning parameter α in Eq. 14.30 is defined as [30]

$$\alpha(t) = \alpha_0 \exp\left(-\left(\frac{t}{\tau_2}\right)\right). \quad (14.31)$$

where α_0 is chosen between 0 and 1 and a time constant τ_2 is set equal to the total number of iterations.

The training of FRGSOM is performed for a fixed number of iterations and the connection weights are updated. The updated weights are more likely to become similar to the input patterns, which are presented to the network during training. After completion of the competitive learning, FRGSOM partitions the 3D input data into clusters (granulation structures) which are then arranged in a topological order [30].

14.4. Experimental Results

The methods of FRGNN and FRGSOM for classification and clustering, respectively, are implemented in the C language and the programs are executed on the Linux operating system installed in a HP computer. The computer is configured with Intel Core i7 CPU 880 @ 3.07GHZ processor and 16 GB RAM. The details of data sets are discussed as follows.

Artificial data: This artificial data [31] contains 250 patterns, two features and five classes. There are fifty patterns in each of the classes. Fig. 14.3 shows the two dimensional plot of the overlapping patterns in the five groups.

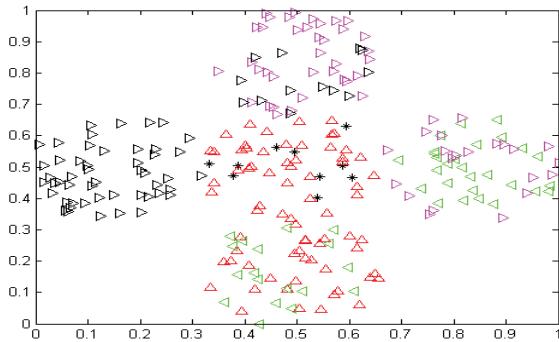


Fig. 14.3. 2D scattered plot of features for artificial data in F₁-F₂ plane.

Haberman's Survival Data:

The data set is based on a study conducted between 1958 and 1970 at the University of Chicago's Billings Hospital on the survival of patients who had undergone surgery for breast cancer. There are 306 patients (patterns) with 3 features for two categories, the patients who survived upto 5 years and the patients who died within 5 years after surgery. The age of the patient at the time of operation, patient's year of operation and the number of positive axillary lymph nodes detected are the values of features. Fig. 14.4 depicts the two dimensional plot of the overlapping patterns in the two categories. The data is downloaded from <https://archive.ics.uci.edu/ml/datasets/Haberman's+Survival>.

Synthetic Pattern1 data: This data [32] consists of 880 patterns with two features. Fig. 14.5 represents the scatter plot of the data for three non convex classes which are linearly non separable in 2D plane.

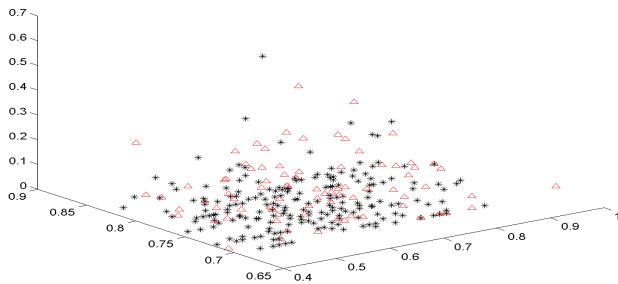


Fig. 14.4. 3D scattered plot of features for Haberman's survival data in F_1 - F_2 - F_3 space.

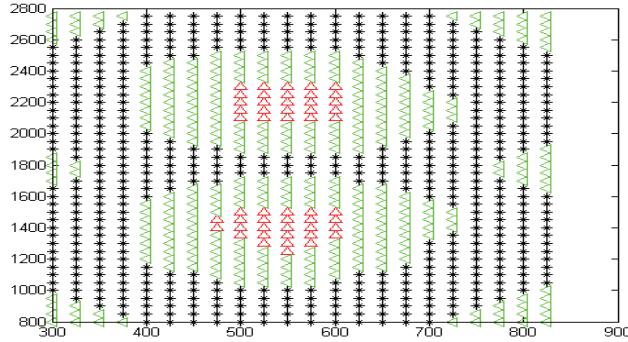


Fig. 14.5. 2D scattered plot of features for synthetic pattern1 data in F_1 - F_2 plane.

14.4.1. Classification Results

The fuzzy rough granular neural network (FRGNN) is used for classifying the real life data sets. During training the network, 5 folds cross validation designed with randomized sampling is used. Out of 5 folds, 4 folds of data, selected randomly from each of the classes, are considered as training set and the remaining one fold is used as test set. The classification process is repeated for 5 times considering 4 folds of training set and 1 fold of test set and the average of the classification accuracies is reported.

Before providing the classification results of test set, the configuration of

FRGNN based on granular input features, knowledge extraction about the data and fuzzy target vector is explained as follows. For defining granular input vector, the data is transformed into 3D granular feature space using Eq. 14.3. The values of center and scaling factor of π membership function along each of the 3D granular features, *low*, *medium* and *high*, are chosen using Eq. 14.6. The procedure for choosing these values is described in Section 14.2.2.1. The fuzzy output vector, in terms of membership value and zeros, for every input pattern is defined using Eq. 14.9 (see Section 14.2.3.1).

Knowledge extraction procedure: Here, we explain the knowledge extracting procedure of FRGNN for synthetic pattern1 data as an example. A 3D granular training set is presented to a decision table $S = (U, \mathcal{A} \cup \{d\})$. It is then divided into 3 decision tables $S_l = (U_l, \mathcal{A} \cup \{d\})$, $l = 1, 2$ and 3 as there are 3 number of classes in the data. Based on the decision table S_l , we use the procedure in Steps 1-5, (see Section 14.2.4.3) for extracting the knowledge about the training set. The resultant knowledge is encoded into the network as its initial connection weights. For example, the initial weights between input nodes & hidden nodes and hidden nodes & output nodes of FRGNN for the synthetic pattern1 data are shown in Table 14.1.

Table 14.1. Initial connection weights of FRGNN for synthetic pattern1 data.

Input to Hidden Layer(w_{jl})			Hidden to Output Layer(w_{lk})		
0.295	0.294	0.279	0.291	0.294	0.290
0.288	0.293	0.278	0.072	0.083	0.104
0.288	0.294	0.279	0.283	0.286	0.284
0.288	0.293	0.279			
0.288	0.288	0.305			
0.294	0.287	0.278			

Performance of FRGNN: The average classification results of 5 folds for the three data sets are compared with the related fuzzy multilayer perceptron (FMLP), K-nearest neighbor (KNN) and Naive Bayes method. The FRGNN becomes fuzzy multilayer perceptron (FMLP) when the initial weights are chosen randomly within 0 to 1. The number of nodes in the input layer for both FRGNN and FMLP is chosen equal to the three dimensional granular input vector. The number of nodes in the output layer is equal to the number of classes in the data. The number of hidden nodes

for FRGNN is chosen equal to the number of decision tables (see Section 14.2.4.3). For FMLP, the number of hidden nodes is varied from 1 to 10 in steps of 1 and the number for which the FMLP provides the highest accuracy is selected. The connection weights of FRGNN are initialized by the dependency factors of features and, the random numbers within 0 to 1 are encoded into FMLP as its initial weights. For a particular number of iterations t , we perform the training process for different values of learning rate η & momentum term α within 0 and 1 by varying them in steps of 0.1. The value of t is chosen by varying it within 100 to 1500 in steps of 100. Hence, the appropriate values of the parameters η , α and t are obtained by trial and error process. The value of 'K' in KNN is chosen as the square root of the number of patterns in the training set. The values of parameters for Naive Bayes method are not specified as the method automatically chose the parameter values.

The results of FRGNN for all the data sets, as compared to FMLP, KNN and Naive Bayes method, are shown in Table 14.2. For all the data sets, the value of t is set equal to 1500 for FRGNN and FMLP. As an example, for one fold of training set and test set, the values of parameters α and η for FRGNN and FMLP are shown in the last column of the table. The number of nodes in the hidden layer of FMLP and the value of K in KNN are provided in parenthesis of the table.

It is clear from Table 14.2 that the performance of FRGNN, in terms of average classification accuracy (99.60%), is superior to FMLP (99.20%), KNN (97.60%) and Naive Bayes (97.20%). Similarly observation can be also made for Haberman's survival data. The results indicate that the FRGNN generalizes the patterns arising in the overlapping regions (see Figs. 14.3 and 14.4) better than the remaining algorithms.

Although the synthetic pattern1 data has concave and linearly non separable classes (see Fig. 14.5), FRGNN has achieved the best average classification accuracy. It can be stated from the results that the FRGNN is an efficient method for handling the data with linearly non separable and concave classes as well as overlapping pattern classes.

We also compared the squared errors of FRGNN and FMLP for all the five folds of test set for all the data sets. Figure 14.6 shows the variation of the error values with the increasing number of iterations for the FRGNN and FMLP for Haberman's survival data.

The errors correspond to that fold of test set for which the best accuracy is achieved. It is clear from the figure that the error decreases with increasing number of iterations. However the error curve for FRGNN is

Table 14.2. Classification accuracies for different algorithms and data sets.

Data	Algorithm	No. of folds					Average results	Parameters
		fold1	fold2	fold3	fold4	fold5		
Artificial	FRGNN	100.00	100.00	100.00	100.00	98.00	99.60	$\alpha = 0.6, \eta = 0.5$
	FMLP(6)	100.00	100.00	100.00	100.00	96.00	99.20	$\alpha = 0.06, \eta = 0.09$
	KNN	100.00	96.00	98.00	96.00	98.00	97.60	
	Naive Bayes	100.00	98.00	98.00	94.00	96.00	97.20	
Haberman's Survival	FRGNN	67.21	78.69	75.41	80.33	75.41	75.41	$\alpha = 0.9, \eta = 0.06$
	FMLP(2)	68.85	67.21	75.41	78.69	73.77	72.79	$\alpha = 0.09, \eta = 0.05$
	KNN	68.85	72.13	75.41	78.68	75.40	74.09	
	Naive Bayes	73.77	65.57	75.40	24.59	44.26	56.72	
Synthetic Pattern1	FRGNN	83.91	88.51	84.48	86.21	86.21	85.86	$\alpha = 0.06, \eta = 0.06$
	FMLP(3)	85.63	90.23	85.63	78.24	85.06	84.96	$\alpha = 0.92, \eta = 0.062$
	KNN	85.63	85.63	79.31	87.35	83.33	84.25	
	Naive Bayes	74.21	74.13	75.28	68.39	66.67	71.84	

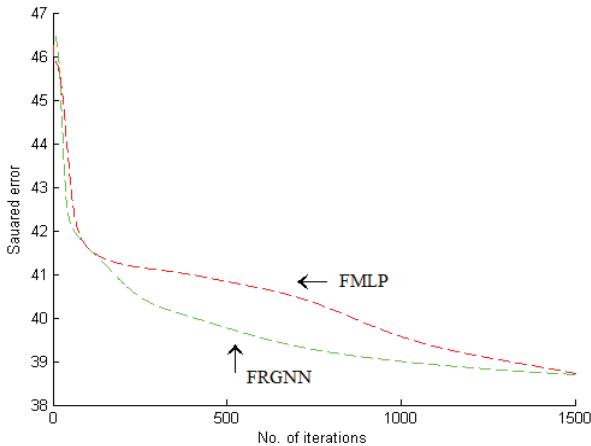


Fig. 14.6. Comparison of squared errors of FRGNN and FMLP for Haberman's survival data.

much lower than that of FMLP, when the number of iterations is within 300 to 1500. The initial connection weights of FRGNN enables it to carry out the learning process better than that those of FMLP by reducing the search space for solution. Hence, the FRGNN achieves the best accuracy.

14.4.2. Clustering Results

The fuzzy rough granular self-organizing map (FRGSOM) is used to perform clustering on three real life data sets, artificial, Haberman's survival and synthetic pattern1. The FRGSOM consists of the input layer and output layer. The input layer contains the number of nodes equal to the number of features in the data. In the output layer, the number of nodes is set equal to c , representing the user defined number of clusters. The initial connection weights of FRGSOM is defined using dependency values of features. The FRGSOM is trained using the competitive learning of self-organizing map (see Section 14.3.2).

The results of FRGSOM is compared with self-organizing map [30], rough possibilistic c -means (RPCM) [33] and c -medoids, in terms of cluster evaluation measures like β -index, DB-index and Dunn-index and fuzzy rough entropy (FRE). The results are provided in Table 14.3. For FRGSOM and SOM, the values of iteration t and learning parameter α_0 are provided in the last column of the table. While the value of threshold Tr

& possibilistic constant w are provided for RPCM, for c -medoids the value of c is shown in the last column of the table. In RPCM, w effects the possibilistic membership of the patterns in a cluster. The value of c is chosen as the same for SOM and FRGSOM.

Table 14.3. Clustering results of all the algorithms, in terms of β -index, DB-index, Dunn-index and FRE, for all the data sets.

Data	Algorithm	β index	DB index	Dunn index	FRE	Para -meters
Artificial	FRGSOM	3.964	1.177	1.183	0.124	$\alpha_0 = 0.09, t = 500$
	SOM	1.551	4.282	0.336	0.147	$\alpha_0 = 0.85, t = 1600$
	RPCM	5.249	1.225	1.000	0.136	$Tr = 0.05, w = 0.35$
	c -medo.	2.665	1.828	0.563	0.161	$c = 5$
Haberman's Survival	FRGSOM	1.695	1.175	1.658	0.238	$\alpha_0 = 0.21, t = 200$
	SOM	1.575	1.065	1.520	0.249	$\alpha_0 = 0.00001, t = 1500$
	RPCM	1.693	1.129	1.464	0.280	$Tr = 0.05, w = 0.05$
	c -medo.	1.594	1.123	1.435	0.296	$c = 2$
Synthetic Pattern1	FRGSOM	2.124	2.029	0.639	0.180	$\alpha_0 = 0.1, t = 2000$
	SOM	3.444	1.838	0.811	0.208	$\alpha_0 = 0.000092, t = 70$
	RPCM	2.727	1.025	0.768	0.221	$Tr = 0.085, w = 0.07$
	c -medo.	2.740	0.902	0.940	0.177	$c = 3$

It is evident from Table 14.3 that the results of artificial data for FRGSOM is better than SOM, RPCM & c -medoids in terms of DB-index, Dunn-index & FRE except β -index, where the FRGSOM is the second best and the RPCM is the best. Note that lower values of FRE & DB-index and higher values of β -index & Dunn-index indicate that the output clusters are better.

For Haberman's survival data, the FRGSOM achieves the best results in terms of β -index, Dunn-index & FRE except DB-index where the FRGSOM is the third best.

For synthetic pattern1 data, it can be seen from Table 14.3 that the performance of FRGSOM is inferior to the remaining algorithms using the most of the clustering measures. The reason for inferior performance of FRGSOM in this case is that synthetic pattern1 data does not have overlapping class boundaries that are in non convex shape. Further, the performance of FRGSOM, which is useful for overlapping patterns, is inferior to SOM for all the indices as the former uses 3D granular input, instead of actual input like latter one. It can be concluded from these results that the FRGSOM efficiently partitions the data whenever there are overlapping patterns.

14.5. Conclusion

The present investigation describes an integration of fuzzy sets and fuzzy rough sets with neural networks in granular computing framework and also demonstrates the effectiveness of the hybrid methods on different types of data. Two methods, fuzzy rough granular neural network (FRGNN) and granular self-organizing map (FRGSOM) are described for performing classification and clustering, respectively. The FRGNN is formulated by incorporating 3D granular input vector, fuzzy initial connection weights and target vector (defined in terms membership values) into multi-layer perceptron. The learning process of FRGNN is performed using the gradient decent method. The FRGNN is compared with fuzzy MLP (FMLP), Naive Bayes and KNN where the first two methods are known for handling overlapping classes and the last method is efficient for handling non convex & non linear classes by generating piecewise linear boundaries. The classification accuracies of FRGNN are found to be better than the remaining algorithms. As the FRGNN incorporates the fuzzy information granulation at the input level & initial connection weights level and the membership values at the output level, it achieves the best classification results. Hence, it can be stated that the FRGNN is an effective algorithm for classifying any data having linearly non separable classes, non convex class shape and overlapping regions between class boundaries.

The performance of FRGSOM is compared with rough possibilistic c -means (RPCM), self-organizing map (SOM) and c -medoids. RPCM is a recent method for clustering data with overlapping patterns, SOM is a classical clustering technique that maps n dimensional input space into two dimensional array such that the output clusters are arranged in a topological order (organization of the output clusters corresponds to the neighborhood relations between them) and c -medoids is also a classical method for clustering data with convex class shape. In most of the cases for the data sets with overlapping patterns, FRGSOM is seen to be superior to others in terms of β -index, DB-index, Dunn-index and fuzzy rough entropy (FRE).

References

- [1] S. K. Pal, Granular mining and rough-fuzzy pattern recognition: A way to natural computation, *IEEE Intelligent Informatics Bulletin*. **13**(1), 3–13 (2012).
- [2] S. K. Pal and S. K. Meher, Natural computing: A problem solving paradigm

- with granular information processing, *Applied Soft Computing*. **13**(9), 3944–3955 (2013).
- [3] J. T. Yao, A. V. Vasilakos, and W. Pedrycz, Granular computing: Perspectives and challenges, *IEEE Transactions on Cybernetics*. **43**(6), 1977–1989 (2013).
 - [4] W. Pedrycz, *Granular computing: Analysis and design of intelligent systems*. Boca Raton: CRC Press, Taylor & Francis Group (2013).
 - [5] Y. Y. Yao, Interpreting concept learning in cognitive informatics and granular computing, *IEEE Transactions on Systems, Man, and Cybernetics, Part-B*. **39**(4), 2009 (855–866).
 - [6] Y. Y. Yao and L. Q. Zhao, A measurement theory view on the granularity of partitions, *Information Sciences*. **213**, 1–13 (2012).
 - [7] L. A. Zadeh, Towards a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic, *Fuzzy Sets and Systems*. **19**, 111–127 (1997).
 - [8] L. A. Zadeh, Fuzzy logic, neural networks, and soft computing, *Communications of the ACM*. **37**, 77–84 (1994).
 - [9] L. A. Zadeh, Fuzzy sets, *Information and Control*. **8**, 338–353 (1965).
 - [10] Z. Pawlak, *Rough sets: Theoretical aspects of reasoning about data*. Kluwer, Netherlands (1991).
 - [11] Z. Pawlak and A. Skowron, Rough sets and boolean reasoning, *Information Sciences*. **177**, 41–73 (2007).
 - [12] A. E. Hassanien, A. Abraham, J. F. Peters, G. Schaefer, and C. Henry, Rough sets and near sets in medical imaging: A review, *IEEE Transactions on Information Technology in Biomedicine*. **13**(6), 955–968 (2009).
 - [13] R. Jensen and Q. Shen, New approaches to fuzzy-rough feature selection, *IEEE Transactions on Fuzzy Systems*. **17**, 824–838 (2009).
 - [14] C. Cornelis, R. Jensen, G. Hurtado, and D. Slezak, Attribute selection with fuzzy decision reducts, *Information Sciences*. **180**, 209–224 (2010).
 - [15] S. K. Pal and A. Skowron, *Rough fuzzy hybridization: New trends in decision making*. Springer Verlag, Singapore (1999).
 - [16] S. K. Pal, S. Mitra, and L. Lamport, *Neuro-fuzzy pattern recognition: Methods in soft computing*. Wiley, New York (1999).
 - [17] Y. Q. Zhang, M. D. Fraser, R. Gagliano, and A. Kandel, Granular neural networks for numericallinguistic data fusion and knowledge discovery, *IEEE Transactions on Neural Networks*. **11**, 658–667 (2000).
 - [18] Y. Q. Zhang, B. Jin, and Y. Tang, Granular neural networks with evolutionary interval learning, *IEEE Transactions on Fuzzy Systems*. **16**, 309–319 (2008).
 - [19] M. Szczuka, Refining classifiers with neural networks, *International Journal of Computer and Information Sciences*. **16**, 39–56 (2001).
 - [20] M. Banerjee, S. Mitra, and S. K. Pal, Rough fuzzy mlp: Knowledge encoding and classification, *IEEE Transactions Neural Networks*. **9**, 1203–1216 (1998).
 - [21] S. K. Pal, B. Dasgupta, and P. Mitra, Rough self-organizing map, *Applied Intelligence*. **21**, 289–299 (2004).
 - [22] P. Lingras, M. Hogo, and M. Snorek, Interval set clustering of web users using

- modified kohonen self-organizing maps based on the properties of rough sets, *Web Intelligence and Agent Systems.* **46**, 105–119 (1999).
- [23] J. P. Herbert and J. T. Yao, A granular computing frame work for self-organizing maps, *Neurocomputing.* **72**, 2865–2872 (2009).
 - [24] S. Mitra and S. K. Pal, Self-organizing neural network as a fuzzy classifier, *IEEE Transactions on Systems, Man and Cybernetics.* **24**, 385–399 (1994).
 - [25] A. Ganivada, S. Dutta, and S. K. Pal, Fuzzy rough granular neural networks, fuzzy granules and classification, *Theoretical Computer Science.* **412**(42), 5834–5853 (2011).
 - [26] A. Ganivada, S. S. Ray, and S. K. Pal, Fuzzy rough sets, and a granular neural network for unsupervised feature selection, *Neural Networks.* **48**, 91–108 (2013).
 - [27] A. Ganivada, S. S. Ray, and S. K. Pal, Fuzzy rough granular self-organizing map and fuzzy rough entropy, *Theoretical Computer Science.* **466**, 37–63 (2012).
 - [28] S. K. Pal and D. Dutta Majumder, *Fuzzy mathematical approach to pattern recognition.* John Wiley, New York (1986).
 - [29] A. M. Radzikowska and E. E. Kerre, A comparative study of fuzzy rough sets, *Fuzzy Sets and Systems.* **126**(2002), 137–155 (2).
 - [30] T. Kohonen. Self-organizing maps. In *Proc. IEEE*, vol. 78, pp. 1464–1480 (1990).
 - [31] S. Bandyopadhyay and U. Maulik, Nonparametric genetic clustering: comparison of validity, *IEEE Transactions on Systems, Man, and Cybernetics PART C.* **31**(1), 120–125 (2001).
 - [32] S. K. Pal and S. Mitra, Fuzzy versions of kohonen’s net and mlp based classification: Performance evaluation for certain nonconvex decision regions, *Information Sciences.* **76**, 297–337 (1994).
 - [33] P. Maji and S. K. Pal, Rough set based generalized fuzzy c-means algorithm and quantitative indices, *IEEE Transactions on Systems, Man, and Cybernetics, PART-B.* **37**(6), 1529–1540 (2007).

Chapter 15

Fundamentals of Rough-Fuzzy Clustering and Its Application in Bioinformatics

Pradipta Maji¹ and Sushmita Paul²

¹Machine Intelligence Unit
Indian Statistical Institute, Kolkata, India
pmaji@isical.ac.in

²Department of Biology
Indian Institute of Technology
Jodhpur, India
sushmitapaul@iitj.ac.in

Cluster analysis is a technique that divides a given data set into a set of clusters in such a way that two objects from the same cluster are as similar as possible and the objects from different clusters are as dissimilar as possible. In this regard, a hybrid unsupervised learning algorithm, termed as rough-fuzzy c -means, is presented in this chapter. It comprises a judicious integration of the principles of rough sets and fuzzy sets. While the concept of lower and upper approximations of rough sets deals with uncertainty, vagueness, and incompleteness in class definition, the membership function of fuzzy sets enables efficient handling of overlapping partitions. The concept of crisp lower approximation and fuzzy boundary of a class, introduced in rough-fuzzy c -means, enables efficient selection of cluster prototypes. The effectiveness of the rough-fuzzy clustering algorithm, along with a comparison with other clustering algorithms, is demonstrated on grouping functionally similar genes from microarray data, identification of co-expressed microRNAs, and segmentation of brain magnetic resonance images using standard validity indices.

15.1. Introduction

Cluster analysis is one of the important problems related to a wide range of engineering and scientific disciplines such as pattern recognition, machine learning, psychology, biology, medicine, computer vision, web intelligence,

communications, and remote sensing. It finds natural groups present in a data set by dividing the data set into a set of clusters in such a way that two objects from the same cluster are as similar as possible and the objects from different clusters are as dissimilar as possible. Hence, it tries to mimic the human ability to group similar objects into classes and categories [1].

A number of clustering algorithms have been proposed to suit different requirements [1, 2]. One of the most widely used prototype based partitional clustering algorithms is k -means or hard c -means (HCM) [3]. The hard clustering algorithms generate crisp clusters by assigning each object to exactly one cluster. When the clusters are not well defined, that is, when they are overlapping, one may desire fuzzy clusters. In this regard, the problem of pattern classification is formulated as the problem of interpolation of the membership function of a fuzzy set by Bellman *et al.* [4], and thereby, a link with the basic problem of system identification is established. A seminal contribution to cluster analysis is Ruspini's concept of a fuzzy partition [5]. The application of fuzzy set theory to cluster analysis was initiated by Dunn and Bezdek by developing fuzzy ISODATA [6] and fuzzy c -means (FCM) [7] algorithms.

The FCM relaxes the requirement of the HCM by allowing gradual memberships [7]. In effect, it offers the opportunity to deal with the data that belong to more than one cluster at the same time. It assigns memberships to an object those are inversely related to the relative distance of the object to cluster prototypes. Also, it can deal with the uncertainties arising from overlapping cluster boundaries. Although the FCM is a very useful clustering method, the resulting membership values do not always correspond well to the degrees of belonging of the data, and it may be inaccurate in a noisy environment [8, 9]. However, in real data analysis, noise and outliers are unavoidable. To reduce this weakness of the FCM, and to produce memberships that have a good explanation of the degrees of belonging for the data, Krishnapuram and Keller [8, 9] proposed possibilistic c -means (PCM) algorithm, which uses a possibilistic type of membership function to describe the degree of belonging. However, the PCM sometimes generates coincident clusters [10]. Recently, some clustering algorithms have been proposed [11–13], integrating both probabilistic and possibilistic fuzzy memberships.

Rough set theory is a new paradigm to deal with uncertainty, vagueness, and incompleteness. It is proposed for indiscernibility in classification or clustering according to some similarity [14]. Some of the early rough clustering algorithms are those due to Hirano and Tsumoto [15] and

De [16]. Other notable algorithms include rough c -means [17], rough self organizing map [18], and rough support vector clustering [19]. Combining fuzzy sets and rough sets provides an important direction in reasoning with uncertainty. Both fuzzy sets and rough sets provide a mathematical framework to capture uncertainties associated with the data. They are complementary in some aspects [20]. Combining both rough sets and fuzzy sets, several rough-fuzzy clustering algorithms, namely, rough-fuzzy c -means (RFCM) [21], rough-possibilistic c -means (RPCM), rough-fuzzy-possibilistic c -means (RFPCM) [22], and robust rough-fuzzy c -means (rRFCM) [23, 24] have been proposed, where each cluster is represented by a lower approximation and a boundary region.

In this regard, this chapter presents the basic notions in the theory of RFCM algorithm [21, 22], as each of the above mentioned rough-fuzzy clustering algorithms can be devised as a special case of it. The RFCM is based on both rough sets and fuzzy sets, where each cluster consists of two regions, namely, lower approximation and boundary region. While the membership function of fuzzy sets enables efficient handling of overlapping partitions, the concept of lower and upper approximations of rough sets deals with uncertainty, vagueness, and incompleteness in class definition. Each partition is represented by a set of three parameters, namely, a cluster prototype or centroid, a crisp lower approximation, and a fuzzy boundary. The lower approximation influences the fuzziness of the final partition. The cluster prototype or centroid depends on the weighting average of the crisp lower approximation and fuzzy boundary. The effectiveness of the RFCM algorithm, along with a comparison with other clustering algorithms, is demonstrated on grouping functionally similar genes from microarray data, identification of co-expressed microRNAs, and segmentation of brain magnetic resonance images using some standard validity indices.

The structure of the rest of this chapter is as follows. Section 15.2 describes the RFCM algorithm based on the theory of rough sets and fuzzy sets. A few case studies and a comparison with other methods are presented in Sections 15.3, 15.4, and 15.5, respectively, for grouping functionally similar genes from microarray data, identification of co-expressed microRNAs, and segmentation of brain magnetic resonance images. Finally, concluding remarks are given in Section 15.6.

15.2. Rough-Fuzzy C-Means Algorithm

This section presents a hybrid clustering algorithm, termed as the RFCM [21, 22]. It is developed by integrating the merits of both rough sets and fuzzy sets. The RFCM adds the concept of fuzzy membership of fuzzy sets, and lower and upper approximations of rough sets into c -means algorithm. While the membership of fuzzy sets enables efficient handling of overlapping partitions, the rough sets deal with uncertainty, vagueness, and incompleteness in class definition [20].

15.2.1. Objective Function

The RFCM algorithm partitions a set $X = \{x_1, \dots, x_i, \dots, x_n\}$ of n objects into c clusters by minimizing the following objective function

$$J = \sum_{i=1}^c J_i \quad (15.1)$$

$$\text{where } J_i = \begin{cases} \omega \times \mathcal{A}_i + (1 - \omega) \times \mathcal{B}_i & \text{if } \underline{\mathcal{A}}(\beta_i) \neq \emptyset, \mathcal{B}(\beta_i) \neq \emptyset \\ \mathcal{A}_i & \text{if } \underline{\mathcal{A}}(\beta_i) \neq \emptyset, \mathcal{B}(\beta_i) = \emptyset \\ \mathcal{B}_i & \text{if } \underline{\mathcal{A}}(\beta_i) = \emptyset, \mathcal{B}(\beta_i) \neq \emptyset \end{cases} \quad (15.2)$$

$$\mathcal{A}_i = \sum_{x_j \in \underline{\mathcal{A}}(\beta_i)} \|x_j - v_i\|^2; \quad (15.3)$$

$$\text{and } \mathcal{B}_i = \sum_{x_j \in B(\beta_i)} (\mu_{ij})^m \|x_j - v_i\|^2; \quad (15.4)$$

where $\underline{\mathcal{A}}(\beta_i)$ and $B(\beta_i)$ represent the lower and boundary regions of cluster β_i , respectively, $\overline{\mathcal{A}}(\beta_i) = \{\underline{\mathcal{A}}(\beta_i) \cup B(\beta_i)\}$ is the upper approximation of β_i , v_i is the centroid of cluster β_i , $\mu_{ij} \in [0, 1]$ is the probabilistic membership of the pattern x_j to cluster β_i , $m \in (1, \infty)$ is the fuzzifier, and the parameters ω and $(1 - \omega)$ correspond to the relative importance of lower and boundary regions. Note that μ_{ij} has the same meaning of membership as that in the FCM. Solving (15.1) with respect to μ_{ij} , we get

$$\mu_{ij} = \left(\sum_{k=1}^c \left(\frac{d_{ij}^2}{d_{kj}^2} \right)^{\frac{1}{m-1}} \right)^{-1}; \text{ where } d_{ij}^2 = \|x_j - v_i\|^2. \quad (15.5)$$

In the RFCM, each cluster is represented by a centroid, a crisp lower approximation, and a fuzzy boundary (Fig. 15.1). The lower approximation influences the fuzziness of final partition. According to the definitions

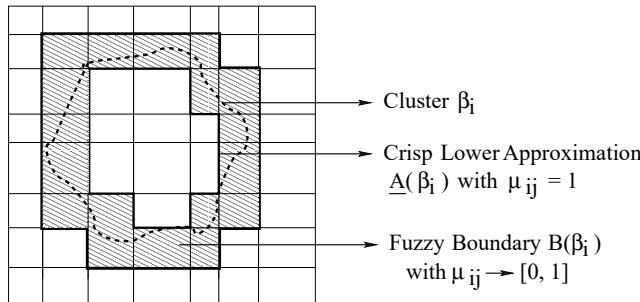


Fig. 15.1. RFCM: each cluster β_i is represented by a crisp lower approximation $\underline{A}(\beta_i)$ and a fuzzy boundary $B(\beta_i)$.

of lower approximation and boundary region of rough sets, if an object $x_j \in \underline{A}(\beta_i)$, then $x_j \notin \underline{A}(\beta_k), \forall k \neq i$, and $x_j \notin B(\beta_i), \forall i$. That is, the object x_j is contained in β_i definitely. Thus, the weights of the objects in lower approximation of a cluster should be independent of other centroids and clusters, and should not be coupled with their similarity with respect to other centroids. Also, the objects in lower approximation of a cluster should have similar influence on the corresponding centroid and cluster. Whereas, if $x_j \in B(\beta_i)$, then the object x_j possibly belongs to β_i and potentially belongs to another cluster. Hence, the objects in boundary regions should have different influence on the centroids and clusters. So, in the RFCM, the membership values of objects in lower approximation are $\mu_{ij} = 1$, while those in boundary region are the same as FCM [(15.5)]. In other word, the RFCM first partitions the data into two classes, namely, lower approximation and boundary. Only the objects in boundary are fuzzified.

15.2.2. Cluster Prototypes

The centroid is calculated based on the weighting average of crisp lower approximation and fuzzy boundary. The centroid calculation for the RFCM is obtained by solving (15.1) with respect to v_i :

$$v_i = \begin{cases} \omega \times v_i + (1 - \omega) \times \tilde{v}_i & \text{if } \underline{A}(\beta_i) \neq \emptyset, B(\beta_i) \neq \emptyset \\ v_i & \text{if } \underline{A}(\beta_i) \neq \emptyset, B(\beta_i) = \emptyset \\ \tilde{v}_i & \text{if } \underline{A}(\beta_i) = \emptyset, B(\beta_i) \neq \emptyset \end{cases} \quad (15.6)$$

$$v_i = \frac{1}{|\underline{A}(\beta_i)|} \sum_{x_j \in \underline{A}(\beta_i)} x_j; \quad (15.7)$$

$$\text{and } \tilde{v}_i = \frac{1}{n_i} \sum_{x_j \in B(\beta_i)} (\mu_{ij})^m x_j; \quad (15.8)$$

$$\text{where } n_i = \sum_{x_j \in B(\beta_i)} (\mu_{ij})^m; \quad (15.9)$$

is the cardinality of $B(\beta_i)$ and $|\underline{A}(\beta_i)|$ represents the cardinality of $\underline{A}(\beta_i)$.

15.2.3. Fundamental Properties

From the above discussions, we can get the following properties of the RFCM algorithm.

- (1) $\bigcup \overline{A}(\beta_i) = U$, U be the set of objects of concern.
- (2) $\underline{A}(\beta_i) \cap \underline{A}(\beta_k) = \emptyset, \forall i \neq k$.
- (3) $\underline{A}(\beta_i) \cap B(\beta_i) = \emptyset, \forall i$.
- (4) $\exists i, k, B(\beta_i) \cap B(\beta_k) \neq \emptyset$.
- (5) $\mu_{ij} = 1, \forall x_j \in \underline{A}(\beta_i)$.
- (6) $\mu_{ij} \in [0, 1], \forall x_j \in B(\beta_i)$.

Let us briefly comment on some properties of the RFCM. The property 2 says that if an object $x_j \in \underline{A}(\beta_i) \Rightarrow x_j \notin \underline{A}(\beta_k), \forall k \neq i$. That is, the object x_j is contained in β_i definitely. The property 3 establishes the fact that if $x_j \in \underline{A}(\beta_i) \Rightarrow x_j \notin B(\beta_i)$, that is, an object may not be in both lower and boundary region of a cluster β_i . The property 4 says that if $x_j \in B(\beta_i) \Rightarrow \exists k, x_j \in B(\beta_k)$. It means an object $x_j \in B(\beta_i)$ possibly belongs to β_i and potentially belongs to other cluster. The properties 5 and 6 are of great importance in computing the objective function J and the cluster prototype v . They say that the membership values of the objects in lower approximation are $\mu_{ij} = 1$, while those in boundary region are the same as the FCM. That is, each cluster β_i consists of a crisp lower approximation $\underline{A}(\beta_i)$ and a fuzzy boundary $B(\beta_i)$.

15.2.4. Convergence Condition

In this section, a mathematical analysis is presented on the convergence property of the RFCM algorithm. According to (15.6), the cluster prototype of the RFCM algorithm is calculated based on the weighting average of the crisp lower approximation and fuzzy boundary when both $\underline{A}(\beta_i) \neq \emptyset$ and $B(\beta_i) \neq \emptyset$, that is,

$$v_i = \omega \times v_i + (1 - \omega) \times \tilde{v}_i \quad (15.10)$$

where v_i and \tilde{v}_i are given by (15.7) and (15.8), respectively. In the RFCM, an object does not belong to both lower approximation and boundary region of a cluster. Hence, the convergence of v_i depends on the convergence of v_i and \tilde{v}_i . Both (15.7) and (15.8) can be rewritten as

$$(|\underline{A}(\beta_i)|)v_i = \sum_{x_j \in \underline{A}(\beta_i)} x_j; \quad (15.11)$$

$$(n_i)\tilde{v}_i = \sum_{x_j \in B(\beta_i)} (\mu_{ij})^m. \quad (15.12)$$

Hence, (15.11) and (15.12) represent a set of linear equations in terms of v_i and \tilde{v}_i , respectively, if μ_{ij} is kept constant. A simple way to analyze the convergence property of the algorithm is to view both (15.7) and (15.8) as the Gauss-Seidel iterations for solving the set of linear equations. The Gauss-Seidel algorithm is guaranteed to converge if the matrix representing each equation is diagonally dominant [25]. This is a sufficient condition, not a necessary one. The iteration may or may not converge if the matrix is not diagonally dominant [25]. The matrix corresponding to (15.7) is given by

$$A = \begin{bmatrix} |\underline{A}(\beta_1)| & 0 & \cdots & \cdots & 0 \\ 0 & |\underline{A}(\beta_2)| & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \cdots & |\underline{A}(\beta_c)| \end{bmatrix} \quad (15.13)$$

where $|\underline{A}(\beta_i)|$ represents the cardinality of $\underline{A}(\beta_i)$. Similarly, the matrix corresponding to (15.8) is given by

$$\tilde{A} = \begin{bmatrix} n_1 & 0 & \cdots & \cdots & 0 \\ 0 & n_2 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \cdots & n_c \end{bmatrix} \quad (15.14)$$

$$\text{where } n_i = \sum_{x_j \in B(\beta_i)} (\mu_{ij})^m$$

which represents the cardinality of $B(\beta_i)$. For both A and \tilde{A} to be diagonally dominant, we must have

$$|\underline{A}(\beta_i)| > 0 \text{ and } n_i > 0. \quad (15.15)$$

This is the sufficient condition for matrices A and \tilde{A} to be diagonally dominant. Under this condition, the iteration would converge if (15.7) and

(15.8) were applied repetitively with μ_{ij} kept constant. In practice, (15.5) and (15.6) are applied alternatively in the iterations. The condition in (15.15) is still correct according to Bezdek's convergence theorem of the FCM [26] and Yan's convergence analysis of the fuzzy curve-tracing algorithm [27]. Both the matrices A and \tilde{A} are also the Hessian (second order derivative) of \mathcal{A}_i and \mathcal{B}_i ((15.1)) with respect to v_i and \tilde{v}_i , respectively. As both A and \tilde{A} are diagonally dominant, all their eigenvalues are positive. Also, the Hessian of \mathcal{B}_i with respect to μ_{ij} can be easily shown to be diagonal matrix and are positive definite. Hence, according to the theorem derived by Bezdek [26] and the analysis done by Yan [27], it can be concluded that the RFCM algorithm converges, at least along a subsequence, to a local optimum solution as long as the condition in (15.15) is satisfied. Intuitively, the objective function J ((15.1)) reduces in all steps corresponding to (15.5) and (15.6), so the compound procedure makes the function J descent strictly.

15.2.5. Details of the Algorithm

Approximate optimization of J , that is, (15.1), by the RFCM algorithm is based on Picard iteration through (15.5) and (15.6). This type of iteration is called alternating optimization. The process starts by initializing c objects as the centroids of the c clusters. The fuzzy memberships of all objects are calculated using (15.5).

Let $\mu_i = (\mu_{i1}, \dots, \mu_{ij}, \dots, \mu_{in})$ be the fuzzy cluster β_i associated with the centroid v_i . After computing μ_{ij} for c clusters and n objects, the values of μ_{ij} for each object x_j are sorted and the difference of two highest memberships of x_j is compared with a threshold value δ . Let μ_{ij} and μ_{kj} be the highest and second highest memberships of x_j . If $(\mu_{ij} - \mu_{kj}) > \delta$, then $x_j \in \underline{A}(\beta_i)$, otherwise $x_j \in B(\beta_i)$ and $x_j \in B(\beta_k)$. After assigning each object in lower approximations or boundary regions of different clusters based on δ , memberships μ_{ij} of the objects are modified. The values of μ_{ij} are set to 1 for the objects lying in lower approximations, while those in boundary regions are remain unchanged. The new centroids of the clusters are calculated as per (15.6). The main steps of the RFCM algorithm proceed as follows:

- (1) Assign initial centroids v_i , $i = 1, 2, \dots, c$. Choose values for fuzzifier \bar{m} , and thresholds ϵ and δ . Set iteration counter $t = 1$.
- (2) Compute μ_{ij} by (15.5) for c clusters and n objects.
- (3) If μ_{ij} and μ_{kj} are the two highest memberships of x_j and $(\mu_{ij} - \mu_{kj}) \leq \delta$,

then $x_j \in B(\beta_i)$ and $x_j \in B(\beta_k)$, otherwise, $x_j \in \underline{A}(\beta_i)$.

- (4) Modify μ_{ij} considering lower approximation and boundary region for c clusters and n objects.
- (5) Compute new centroid as per (15.6).
- (6) Repeat steps 2 to 5, by incrementing t , until $|\mu_{ij}(t) - \mu_{ij}(t-1)| > \epsilon$.

The performance of the RFCM depends on the value of δ , which determines the class labels of all the objects. In other word, the RFCM partitions the data set into two classes, namely, lower approximation and boundary region, based on the value of δ . In practice, we find that the following definition works well:

$$\delta = \frac{1}{n} \sum_{j=1}^n (\mu_{ij} - \mu_{kj}) \quad (15.16)$$

where n is the total number of objects, μ_{ij} and μ_{kj} are the highest and second highest memberships of x_j . Hence, the value of δ represents the average difference of two highest memberships of all the objects in the data set. A good clustering procedure should make the value of δ as high as possible. The value of δ is, therefore, data dependent. If $\delta = 1$, then the RFCM is equivalent to the FCM algorithm [7]. On the other hand, the RFCM boils down to the HCM for $\delta = 0$. Fig. 15.2 shows the membership values of all the objects to two Gaussian-distributed clusters for the FCM and RFCM considering $\delta = 0.95$. The different shape of the memberships, especially for objects near to centers and objects far from the separating line, is apparent.

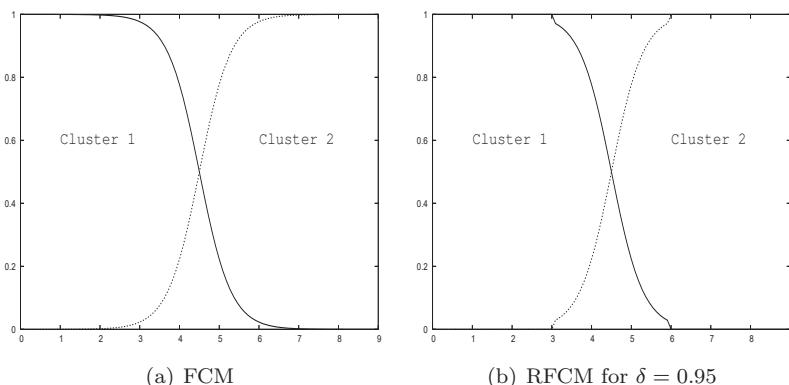


Fig. 15.2. Plot of memberships for FCM and RFCM algorithms.

In this regard, it should be noted that the weighted sum [$a \mu_{ij} + (1 - a) \nu_{ij}$] of probabilistic memberships μ_{ij} and possibilistic memberships ν_{ij} in the boundary region of a cluster turns the RFCM algorithm of Maji and Pal [21] into the RFPCM [22], while the RFPCM reduces to the RPCM algorithm for $a = 0$ [22]. On the other hand, if the lower approximation of each cluster becomes possibilistic in nature, instead of crisp, the RFCM is generalized to the rRFCM algorithm of Maji and Paul [23, 24].

15.3. Grouping Functionally Similar Genes from Microarray

A microarray gene or mRNA expression data set can be represented by an expression table, where each row corresponds to one particular gene, each column to a sample or time point, and each entry of the matrix is the measured expression level of a particular gene in a sample or time point, respectively. However, the large number of genes and the complexity of biological networks greatly increase the challenges of comprehending and interpreting the resulting mass of data, which often consists of millions of measurements. A first step toward addressing this challenge is the use of clustering techniques, which is essential in the pattern recognition process to reveal natural structures and identify interesting patterns in the underlying data. To understand gene function, gene regulation, cellular processes, and subtypes of cells, clustering techniques have proven to be helpful. The co-expressed genes or mRNAs, that is, genes with similar expression patterns, can be clustered together with similar cellular functions. This approach may further provide understanding of the functions of many genes for which information has not been previously available [28].

Different clustering techniques such as hierarchical clustering [29], k -means or HCM algorithm [30], self organizing map (SOM) [31], graph theoretical approaches [32–35], model based clustering [36–39], and density based approach [40] have been widely applied to find groups of co-expressed genes from microarray data. A comprehensive survey on various gene clustering algorithms can be found in [41, 42]. One of the main problems in gene expression data analysis is uncertainty. Some of the sources of this uncertainty include incompleteness and vagueness in cluster definitions. Also, the gene expression data are often highly connected, and the clusters may be highly overlapping with each other [40]. Moreover, gene expression data often contains a huge amount of noise due to the complex procedures of microarray experiments [43]. In this regard, this section presents the application of the RFCM [22] for grouping functionally similar genes from

microarray data. It also reports the comparative performance analysis of the RFCM and different clustering algorithms such as HCM [30], FCM [7], SOM [31], and cluster identification via connectivity kernels (CLICK) [34] on several mRNA expression data. The initialization method reported next is used to select c initial cluster prototypes for different c -means algorithms, while the value of c for each data set is decided by using the CLICK algorithm [34].

15.3.1. Selection of Initial Cluster Prototype

A limitation of the c -means algorithm is that it can only achieve a local optimum solution that depends on the initial choice of the cluster prototype. Consequently, computing resources may be wasted in that some initial centers get stuck in regions of the input space with a scarcity of data points and may therefore never have the chance to move to new locations where they are needed. To overcome this limitation of the c -means algorithm, next a method is presented to select initial cluster prototype. It enables the algorithm to converge to an optimum or near optimum solutions [43].

The normalized Euclidean distance is used for the selection of initial clusters, where the Euclidean distance between two objects x_i and x_j is defined as

$$d_E(x_i, x_j) = \sqrt{\sum_{k=1}^m (x_{ik} - x_{jk})^2}. \quad (15.17)$$

Based on the normalized Euclidean distance, next a method is described for selecting initial cluster centers. The main steps of this method proceed as follows:

- (1) For each object x_i , calculate normalized $d_E(x_j, x_i)$ between itself and the object x_j , $\forall_{j=1}^n$.
- (2) Calculate the similarity score between objects x_i and x_j

$$S(x_j, x_i) = \begin{cases} 1 & \text{if normalized } d_E(x_j, x_i) \leq \lambda \\ 0 & \text{otherwise.} \end{cases} \quad (15.18)$$

- (3) Calculate the total number of similar objects of x_i as

$$N(x_i) = \sum_{j=1}^n S(x_j, x_i). \quad (15.19)$$

- (4) Sort n objects according to their values of $N(x_i)$ such that $N(x_1) > N(x_2) > \dots > N(x_n)$.

- (5) If $N(x_i) > N(x_j)$ and $d_E(x_j, x_i) \leq \lambda$, then x_j cannot be considered as an initial center, resulting in a reduced object set to be considered for initial cluster centers.

Finally, c initial centers are selected from the reduced set as potential initial centers. The main motivation of introducing this initialization method lies in identifying different dense regions present in the data set. The identified dense regions ultimately lead to discovery of natural groups present in the data set. The whole approach is, therefore, data dependent [43].

15.3.2. Optimum Values of Different Parameters

The threshold λ in (15.18) plays an important role to generate the initial cluster centers. It controls the degree of similarity among the objects present in a data set. In effect, it has a direct influence on the performance of the initialization method. Also, the performance of the RFCM algorithm depends on weight parameter ω . Since the objects lying in lower approximation definitely belong to a cluster, they are assigned a higher weight ω compared to $(1 - \omega)$ of the objects lying in boundary regions. On the other hand, the performance of the RFCM significantly reduces when $\omega \simeq 1.00$. In this case, since the clusters cannot see the objects of boundary regions, the mobility of the clusters and the centroids reduces. As a result, some centroids get stuck in local optimum. Hence, to have the clusters and the centroids a greater degree of freedom to move, $0 < (1 - \omega) < \omega < 1$.

Let $\mathcal{S} = \{\lambda, \omega\}$ be the set of parameters and $\mathcal{S}^* = \{\lambda^*, \omega^*\}$ is the set of optimal parameters. To find out the optimum set \mathcal{S}^* , containing optimum values of λ^* and ω^* , the Silhouette index [44] is used. Let a gene $x_i \in \beta_r$, $i = 1, \dots, n_r$ and n_r is the cardinality of cluster β_r . For each gene x_i let a_i be the average distance between gene x_i and rest of the genes of β_r , that is,

$$a_i = d_{\text{avg}}(x_i, \beta_r - \{x_i\}) \quad (15.20)$$

where $d_{\text{avg}}(\cdot, \cdot)$ denotes the average distance measure between a gene and a set of genes. For any other cluster $\beta_p \neq \beta_r$, let $d_{\text{avg}}(x_i, \beta_p)$ denote the average distance of gene x_i to all genes of β_p . The scalar b_i is the smallest of these $d_{\text{avg}}(x_i, \beta_p)$, $p = 1, \dots, c, p \neq r$, that is,

$$b_i = \min_{p=1, \dots, c, p \neq r} \{d_{\text{avg}}(x_i, \beta_p)\}. \quad (15.21)$$

The Silhouette width of gene x_i is then defined as

$$s(x_i) = \frac{b_i - a_i}{\max\{b_i, a_i\}} \quad (15.22)$$

where $-1 \leq s(x_i) \leq 1$. The value of $s(x_i)$ close to 1 implies that the distance of gene x_i from the cluster β_r where it belongs is significantly less than the distance between x_i and its nearest cluster excluding β_r , which indicates that x_i is well clustered. On the other hand, the value of $s(x_i)$ close to -1 implies that the distance between x_i and β_r is significantly higher than the distance between x_i and its nearest cluster excluding β_r , which indicates that x_i is not well clustered. Finally, the values of $s(x_i)$ close to 0 indicate that x_i lies close to the border between the two clusters. Based on the definition of $s(x_i)$, the Silhouette of the cluster β_k ($k = 1, \dots, c$) is defined as

$$S(\beta_k) = \frac{1}{n_k} \sum_{x_i \in \beta_k} s(x_i) \quad (15.23)$$

where n_k is the cardinality of the cluster β_k . The global Silhouette index for a data set is defined as

$$S_c = \frac{1}{c} \sum_{k=1}^c S(\beta_k) \quad (15.24)$$

where $S_c \in [-1, 1]$. Also, the higher the value of S_c , the better the corresponding clustering is. For each microarray data set, the value of threshold λ is varied from 0.00 to 0.15, while the value of weight parameter ω is varied from 0.51 to 0.99. The optimum values of λ^* and ω^* for each data set are obtained using the following relation:

$$\mathcal{S}^* = \arg \max_{\mathcal{S}} \{S_c\}. \quad (15.25)$$

In the present research work, publicly available six microarray mRNA expression data sets with accession numbers GDS608, GDS759, GDS1013, GDS1550, GDS2002, and GDS2003, are used to compare the performance of different clustering methods, which are downloaded from “Gene Expression Omnibus” (<http://www.ncbi.nlm.nih.gov/geo/>). The optimum values of λ obtained using (15.25) are 0.14, 0.11, 0.13, 0.07, 0.13, and 0.15 for GDS608, GDS759, GDS1013, GDS1550, GDS2002, and GDS2003 data sets, respectively, while the optimum values of ω obtained using (15.25) are 0.95 for GDS1013 and GDS2002, and 0.99 for rest four data sets.

15.3.3. Performance of Different Clustering Algorithms

This section presents the comparative performance analysis of different gene clustering algorithms with respect to Silhouette index [44], Davies-Bouldin (DB) index [45], and Dunn index [45]. A good clustering algorithm should

make the values of Silhouette and Dunn indices as high as possible and that of DB index as low as possible.

In order to establish the importance of crisp lower approximation of the RFCM [21], extensive experimentation is carried out on six yeast microarray data sets. Results and subsequent discussions are presented in Table 15.1, along with the performance of both HCM [30] and FCM [46], for optimum values of λ . The bold value in Table 15.1 signifies the best value. All the results reported in Table 15.1 establish the fact that the RFCM algorithm is superior to other c -means clustering algorithms. The best performance of the RFCM, in terms of Silhouette, DB, and Dunn indices, is achieved due to the fact that the probabilistic membership function of the RFCM handles efficiently overlapping gene clusters; and the concept of crisp lower approximation and probabilistic boundary of the RFCM algorithm deals with uncertainty, vagueness, and incompleteness in cluster definition.

Table 15.1. Comparative Performance Analysis of Different C-Means Algorithms.

Index	Method	GDS608	GDS759	GDS1013	GDS1550	GDS2002	GDS2003
Silhouette	HCM	0.083	0.151	0.450	0.422	0.149	0.139
	FCM	0.006	0.029	0.243	0.321	-	-0.377
	RFCM	0.140	0.210	0.468	0.426	0.225	0.218
DB	HCM	1.927	1.833	0.407	0.409	1.811	1.470
	FCM	18.914	11.571	1.208	0.926	-	14.583
	RFCM	1.137	0.872	0.444	0.465	0.910	0.669
Dunn	HCM	0.261	0.084	0.109	0.090	0.028	0.073
	FCM	0.000	0.000	0.008	0.025	-	0.000
	RFCM	0.363	0.212	0.323	1.012	0.170	0.979

In order to establish the superiority of the RFCM algorithm over two existing gene clustering algorithms, namely, CLICK [34] and SOM [31], extensive experimentation is performed on six yeast microarray data sets. Table 15.2 presents the comparative assessment of these three clustering

Table 15.2. Comparative Performance Analysis of Different Algorithms.

Index	Data Sets	GDS608	GDS759	GDS1013	GDS1550	GDS2002	GDS2003
Silhouette	CLICK	-0.040	-0.080	-0.520	-0.490	-0.120	-0.090
	SOM	-0.030	-0.020	0.060	0.150	-0.050	-0.060
	RFCM	0.140	0.210	0.468	0.426	0.225	0.218
DB	CLICK	11.520	27.910	9713.810	525.510	26.700	17.610
	SOM	18.030	19.030	3.400	2.090	13.410	15.220
	RFCM	1.137	0.872	0.444	0.465	0.910	0.669
Dunn	CLICK	0.060	0.020	0.000	0.000	0.030	0.050
	SOM	0.020	0.010	0.000	0.000	0.000	0.010
	RFCM	0.363	0.212	0.323	1.012	0.170	0.979

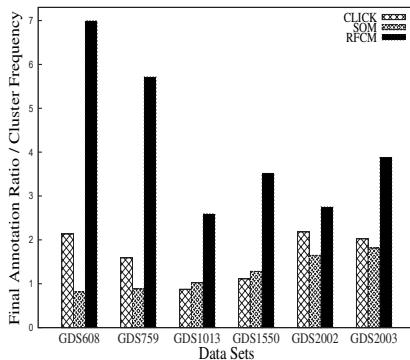
algorithms, in terms of Silhouette index, DB index, and Dunn index, where bold value represents the best value. From the results reported in this table, it can be seen that the RFCM algorithm performs significantly better than both CLICK and SOM, irrespective of microarray data sets and quantitative indices used. Hence, the RFCM algorithm can identify compact groups of co-expressed genes.

15.3.4. Functional Consistency of Clustering Result

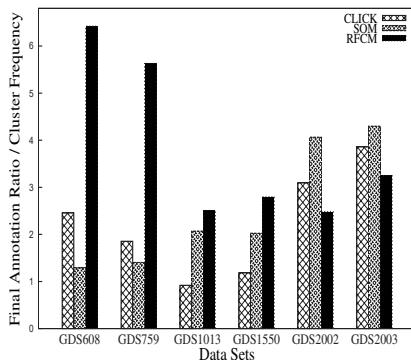
In order to evaluate the functional consistency of the gene clusters produced by different algorithms, the biological annotations of the gene clusters are considered in terms of the gene ontology (GO) [47, 48]. The annotation ratios of each gene cluster in three GO ontologies, namely, biological processes (BP), molecular functions (MF), and cellular components (CC), are calculated using the GO Term Finder [47]. It finds the most significantly enriched GO terms associated with the genes belonging to a cluster. The GO project aims to build tree structures, controlled vocabularies, also called ontologies, that describe gene products in terms of their associated BP, MF, and CC. The GO term is searched in which most of the genes of a particular cluster are enriched. The annotation ratio, also termed as cluster frequency, of a gene cluster is defined as the number of genes in both the assigned GO term and the cluster divided by the number of genes in that cluster. A higher value of annotation ratio indicates that the majority of genes in the cluster are functionally more closer to each other, while a lower value signifies that the cluster contains much more noises or irrelevant genes. After computing the annotation ratios of all gene clusters for a particular ontology, the sum of all annotation ratios is treated as the final annotation ratio. A higher value of final annotation ratio represents that the corresponding clustering result is better than other, that is, the genes are better clustered by function, indicating a more functionally consistent clustering result [24].

Figure 15.3 compares the final annotation ratios obtained using the CLICK, SOM, and RFCM algorithms. From the results reported in Fig. 15.3, it can be seen that the final annotation ratio obtained using the RFCM algorithm is higher than that obtained using both CLICK and SOM in most of cases. For MF and CC ontologies, the RFCM performs better than both CLICK and SOM, irrespective of the data sets used, while the RFCM achieves best performance in four cases out of total six cases of BP ontology. Hence, all the results reported in Fig. 15.3 establish the fact

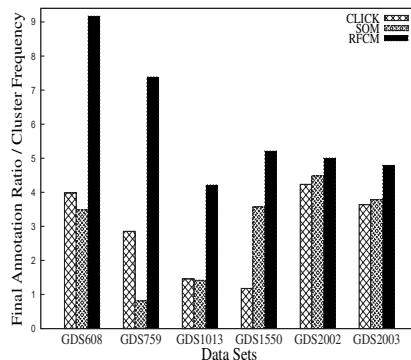
that the majority of genes in a cluster produced by the RFCM algorithm are functionally more closer to each other than those by other algorithms, while the clusters obtained using existing algorithms include much more noises or irrelevant genes.



(a) Molecular Function



(b) Biological Process



(c) Cellular Component

Fig. 15.3. Biological annotation ratios of different algorithms on six gene expression data.

15.4. Clustering Co-Expressed miRNAs

The micro RNAs, also termed as miRNAs, a class of short approximately 22-nucleotide non-coding RNAs found in many plants and animals, often act post-transcriptionally to inhibit gene expression. Hence, miRNAs are

related to diverse cellular processes and regarded as important components of the gene regulatory network [50]. With the advancement of technologies, it is now possible to profile expression of several miRNA's at a time. The miRNA expression profiling has been done to decipher its role in a cellular processes or particular disease [51]. Several miRNAs may suppress the activity or expression of one mRNA and thus may involve in a particular disease or cellular activity [52–54]. Grouping these miRNAs may help in understanding the etiology of the disease or cellular processes. Applying clustering techniques, miRNAs having similar cellular activities can be grouped together. In this background, different clustering algorithms have been used in order to group miRNAs having similar function [23, 49, 55–57].

This section presents the application of the RFCM [22] algorithm for grouping co-expressed miRNAs, along with a comparison with different clustering algorithms such as HCM [30], FCM [7], SOM [31], and CLICK [34] on several miRNA expression data. The initial cluster prototypes for each c -means algorithm are selected using the procedure reported in Section 15.3.1, while the value of c for each data set is decided by using the CLICK algorithm [34]. The four miRNA expression data sets with accession numbers GSE16473, GSE17155, GSE29495, and GSE35074, are used in this study, which are available at “Gene Expression Omnibus”. The optimum values of λ obtained using (15.25) are 0.11, 0.01, 0.12, and 0.05 for GSE16473, GSE17155, GSE29495, and GSE35074 data sets, respectively, while the optimum values of ω obtained using (15.25) are 0.95 for GSE29495 and 0.99 for rest three data sets.

Tables 15.3 and 15.4 present the comparative performance analysis of different clustering algorithms on four miRNA expression data sets with respect to Silhouette, DB, and Dunn indices. From the results reported in Table 15.3, it is seen that the RFCM performs better than other c -means clustering algorithms in most of the cases. Out of total 12 cases, the RFCM provides better results in 8 cases, while the HCM attains it only for 4 cases, irrespective of the cluster validity indices and data sets used. Similarly, the results reported in Table 15.4 confirm that the RFCM performs significantly better than both CLICK and SOM algorithms. While the RFCM attains best results in 11 cases out of total 12 cases, the best result is achieved by the CLICK in only 1 case. All the results reported in Tables 15.3 and 15.4 establish the fact that the RFCM algorithm is able to identify co-expressed miRNA clusters more efficiently than do the existing algorithms.

Table 15.3. Comparative Performance Analysis of Different C-Means Algorithms.

Index	Method	GSE16473	GSE17155	GSE29495	GSE35074
Silhouette	HCM	0.595	0.204	0.696	0.045
	FCM	0.238	0.122	0.519	*
	RFCM	0.417	0.221	0.726	0.088
DB	HCM	0.608	1.544	0.257	3.596
	FCM	4.406	116.478	1.182	*
	RFCM	3.560	1.104	0.454	2.760
Dunn	HCM	0.146	0.570	1.774	0.184
	FCM	0.064	0.003	0.085	*
	RFCM	0.165	0.802	0.561	0.235

Table 15.4. Comparative Performance Analysis of Different Algorithms.

Index	Algorithms	GSE16473	GSE17155	GSE29495	GSE35074
Silhouette	CLICK	0.005	-0.101	-0.634	0.038
	SOM	0.059	-0.112	-0.540	0.009
	RFCM	0.417	0.221	0.726	0.088
DB	CLICK	2.277	13.016	450.689	8.929
	SOM	10.128	39.558	455.345	19.875
	RFCM	3.560	1.104	0.454	2.760
Dunn	CLICK	0.101	0.003	0.000	0.007
	SOM	0.011	0.001	0.000	0.003
	RFCM	0.165	0.802	0.561	0.235

15.5. Segmentation of Brain Magnetic Resonance Images

Segmentation is a process of partitioning an image space into some non-overlapping meaningful homogeneous regions. In medical image analysis for computer-aided diagnosis and therapy, segmentation is often required as a preliminary stage. Segmentation of brain images into three main tissue classes, namely, white matter, gray matter, and cerebro-spinal fluid, is essential for many diagnostic studies. The MRI is a noninvasive diagnostic medical image acquisition tool that has the ability to provide high resolution images with excellent discrimination of soft tissues based on different MR parameters. In general, the brain MR images are piecewise constant with small number of tissue classes, which makes the segmentation procedure much more easier and reliable than the other imaging modalities.

Many image processing techniques have been proposed for MR image segmentation [58]. Most notable among them is thresholding [59, 60], which segments the scalar images by using one or more thresholds. The region-growing [61, 62] is another popular technique for MR image segmentation, which is generally used for delineation of small, simple structure such as

tumors and lesions. However, it is sensitive to noise and partial volume effects. Some algorithms using artificial neural network approach have also been investigated in the MR image segmentation problem [63], either applied in supervised fashion [64] or unsupervised fashion [58, 65]. Deformable models such as snake model [66] have also been widely used in MR image segmentation. Among other techniques, atlas-guided [67, 68], SOM [69], wavelet transform [70, 71], k -means or HCM [72, 73], and graph-cut [74] based approaches are applied in brain MRI for segmentation of various structures.

One of the main problems in MR image segmentation is uncertainty. Some of the sources of this uncertainty include imprecision in computations and vagueness in class definitions. In this background, the possibility concept introduced by the theory of probability, fuzzy set, and rough sets have gained popularity in modeling and propagating uncertainty. They provide mathematical frameworks to capture uncertainties associated with human cognition process. In this regard, the segmentation of MR images using FCM has been reported in [64, 75–78], while the RFCM is used in [21, 22, 79] for segmentation of brain MR images. Another popular framework to model brain tissue classes for segmentation is the probabilistic model, more specifically, the finite Gaussian mixture (FGM) model [80–82].

This section presents the application of rough-fuzzy clustering for segmentation of brain MR images into meaningful tissue classes. To establish the importance of the RFCM algorithm [22] for brain MR image segmentation, extensive experimentation is carried out and the performance of the RFCM is compared with that of HCM, FCM, and FGM model [81]. To analyze the performance of different algorithms, the experimentation is done on some benchmark simulated MR images obtained from “BrainWeb: Simulated Brain Database” (<http://www.bic.mni.mcgill.ca/brainweb/>) and real MR images of “IBSR: Internet Brain Segmentation Repository” (<http://www.cma.mgh.harvard.edu/ibsr/>). For BrainWeb database, the results are reported for different noise levels and intensity inhomogeneity. All the image volumes of BrainWeb and IBSR are of size $217 \times 181 \times 181$ and $256 \times 128 \times 256$, respectively. Before applying any algorithm for segmentation of brain MR images, the brain extraction tool [83] is applied to remove non-brain tissues like skull, scalp and dura, while contaharmonic mean and rough sets based RC2 algorithm [84] is used to remove intensity inhomogeneity artifact or bias field present in MR images. The initial cluster prototypes of different algorithms are obtained using the method proposed by Otsu [85]. The comparative performance analysis is studied

with respect to various segmentation metrics, namely, dice coefficient, sensitivity, and specificity.

To find out the optimum value of weight parameter ω for a particular image of BrainWeb and IBSR database, the procedure reported in Section 15.3.2 is used. For all images of BrainWeb and IBSR databases, the value of ω is varied from 0.51 to 0.99. The optimum value of ω obtained using (15.25) for the RFCM algorithm are 0.99 for 17 images and 0.95 for only one image of BrainWeb database. For IBSR database, the optimum values are 0.99 for 8 images and 0.95 for remaining two images.

Figure 15.4 represents the variation of Silhouette index with respect to different values of ω on four images of BrainWeb and IBSR databases as well as depicts the variation of dice coefficient with respect to weight parameter ω . From the results reported in Fig. 15.4, it can be seen that the values of Silhouette index and dice coefficient vary in similar fashion. The

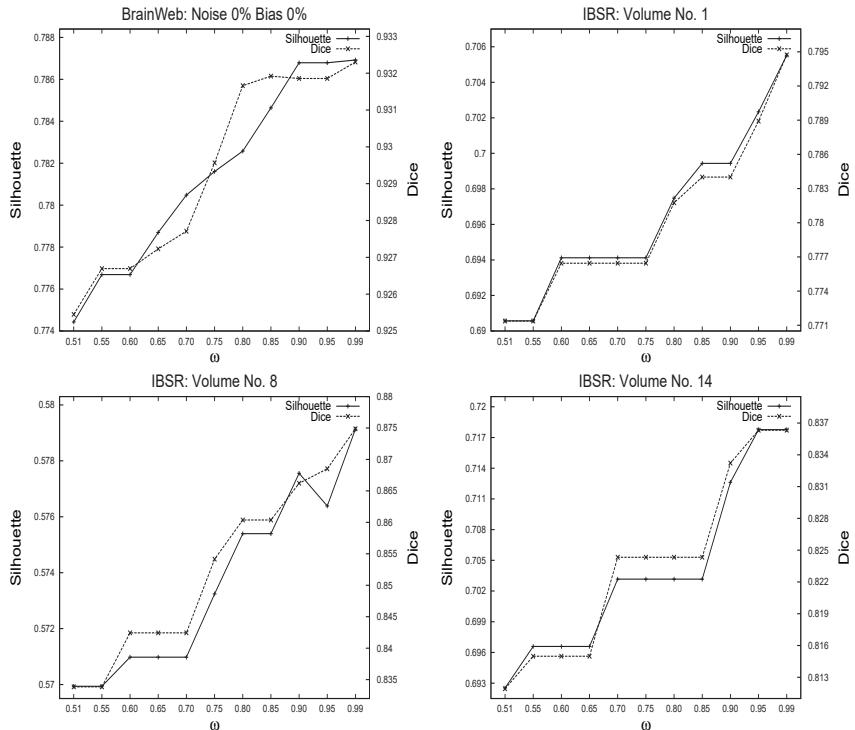


Fig. 15.4. Variation of Silhouette index and dice coefficient for different values of weight parameter ω on BrainWeb and IBSR databases.

Silhouette index is computed based on the generated clusters, while dice coefficient is calculated based on the ground truth segmentation information. Hence, if these two indices are found to vary similarly, the optimum value of Silhouette index would produce optimum value of dice coefficient for the same parameter configuration, thus producing optimum segmentation result. The best dice coefficient obtained from all possible ω values on each data set is compared with the dice coefficient corresponding to optimum ω^* . From all the results reported in Table 15.5, it is visible that the dice coefficient corresponding to ω^* is exactly same with the best dice coefficient in 15 cases out of total 28 cases, while in all other cases they are comparable.

Table 15.5. Dice Coefficients on BrainWeb and IBSR.

Volume	Dice at ω^*	Best Dice	Volume	Dice at ω^*	Best Dice
0-0	0.932	0.932	5-0	0.924	0.925
0-20	0.932	0.934	5-20	0.931	0.931
0-40	0.927	0.928	5-40	0.922	0.923
1-0	0.934	0.934	7-0	0.904	0.905
1-20	0.932	0.932	7-20	0.908	0.909
1-40	0.929	0.929	7-40	0.907	0.907
3-0	0.933	0.935	9-0	0.876	0.876
3-20	0.933	0.934	9-20	0.881	0.882
3-40	0.931	0.932	9-40	0.879	0.880
V1	0.795	0.795	V9	0.852	0.855
V2	0.796	0.798	V10	0.872	0.872
V3	0.768	0.768	V13	0.782	0.782
V5	0.705	0.705	V14	0.836	0.836
V8	0.875	0.875	V17	0.874	0.874

15.5.1. Performance of Different Clustering Algorithms

This section presents the comparative performance analysis of the RFCM algorithm and several clustering algorithms such as HCM, FCM, and FGM. Results are reported in Tables 15.6 and 15.7 on several images of BrainWeb and IBSR databases with respect to three segmentation evaluation indices, namely, dice coefficient, sensitivity, and specificity. The comparative performance analysis is also reported in terms of p-value computed through Wilcoxon signed-rank test (one-tailed). From all the results reported in Tables 15.6 and 15.7, it can be seen that the RFCM provides significantly better segmentation results compared to FGM, irrespective of the quantitative indices used. On the other hand, the RFCM attains significantly better results compared to HCM and FCM with respect to Dice coefficient

Table 15.6. Dice Coefficient for Different Clustering Algorithms.

Vol. No.	Dice				Vol. No.	Dice			
	RFCM	HCM	FCM	FGM		RFCM	HCM	FCM	FGM
0-0	0.932	0.927	0.929	0.894	5-0	0.924	0.925	0.925	0.924
0-20	0.932	0.930	0.930	0.914	5-20	0.931	0.930	0.931	0.930
0-40	0.927	0.929	0.929	0.917	5-40	0.922	0.921	0.921	0.919
1-0	0.934	0.935	0.935	0.909	7-0	0.904	0.904	0.904	0.902
1-20	0.932	0.931	0.931	0.922	7-20	0.908	0.907	0.908	0.907
1-40	0.929	0.929	0.929	0.924	7-40	0.907	0.906	0.906	0.902
3-0	0.933	0.934	0.934	0.932	9-0	0.876	0.868	0.872	0.876
3-20	0.933	0.932	0.933	0.934	9-20	0.881	0.879	0.879	0.882
3-40	0.931	0.931	0.931	0.933	9-40	0.879	0.877	0.879	0.881
V1	0.795	0.785	0.790	0.777	V9	0.852	0.809	0.829	0.848
V2	0.796	0.789	0.792	0.769	V10	0.872	0.844	0.863	0.861
V3	0.768	0.755	0.763	0.755	V13	0.782	0.777	0.780	0.781
V5	0.705	0.699	0.704	0.691	V14	0.836	0.835	0.834	0.834
V8	0.875	0.858	0.866	0.872	V17	0.874	0.866	0.866	0.843
p-value with respect to RFCM						3.6E-05	8.6E-05	3.9E-05	

and specificity, whereas the performance of the HCM and FCM is better, but not significantly (marked in bold), compared to that of the RFCM algorithm with respect to sensitivity.

Figures 15.5, 15.6, 15.7 and 15.8 present the segmented images obtained using different methods, along with the original images of BrainWeb and IBSR data sets and corresponding ground truth images. The first and second images of Figs. 15.5, 15.6, 15.7 and 15.8 show the original and ground truth images, respectively, while remaining images present the segmented images produced by different methods. The third, fourth, fifth, and sixth images of Figs. 15.5, 15.6, 15.7 and 15.8 compare the performance of the RFCM, HCM, FCM, and FGM qualitatively. All the results reported in Tables 15.6 and 15.7, and Figs. 15.5, 15.6, 15.7, and 15.8 establish the fact that the RFCM performs better segmentation than do the existing clustering algorithms, irrespective of the data sets used.

15.6. Conclusion

This chapter presents a rough-fuzzy clustering algorithm, termed as rough-fuzzy c -means (RFCM), integrating judiciously the merits of rough sets and fuzzy c -means algorithm. This formulation is geared towards maximizing the utility of both rough sets and fuzzy sets with respect to knowledge discovery tasks. The effectiveness of the algorithm is demonstrated, along with a comparison with other related algorithms, on a set of real life data sets.

Table 15.7. Sensitivity and Specificity for Different Clustering Algorithms.

Vol. No.	Sensitivity				Specificity			
	RFCM	HCM	FCM	FGM	RFCM	HCM	FCM	FGM
0-0	0.932	0.934	0.935	0.887	0.983	0.982	0.983	0.974
0-20	0.936	0.931	0.930	0.913	0.986	0.985	0.985	0.979
0-40	0.934	0.928	0.928	0.914	0.984	0.984	0.984	0.980
1-0	0.932	0.931	0.931	0.902	0.984	0.983	0.983	0.978
1-20	0.934	0.936	0.936	0.920	0.984	0.984	0.984	0.981
1-40	0.936	0.929	0.928	0.924	0.986	0.985	0.986	0.982
3-0	0.930	0.928	0.927	0.928	0.985	0.984	0.984	0.984
3-20	0.932	0.935	0.935	0.935	0.986	0.985	0.985	0.985
3-40	0.938	0.932	0.932	0.933	0.984	0.983	0.983	0.985
5-0	0.915	0.920	0.919	0.918	0.983	0.983	0.983	0.983
5-20	0.930	0.934	0.934	0.928	0.985	0.984	0.985	0.984
5-40	0.921	0.922	0.922	0.914	0.983	0.983	0.983	0.981
7-0	0.898	0.901	0.901	0.896	0.979	0.979	0.979	0.978
7-20	0.902	0.909	0.909	0.905	0.980	0.980	0.980	0.980
7-40	0.902	0.907	0.907	0.897	0.979	0.979	0.979	0.978
9-0	0.872	0.875	0.875	0.872	0.973	0.972	0.972	0.973
9-20	0.877	0.883	0.883	0.881	0.974	0.974	0.974	0.974
9-40	0.875	0.881	0.881	0.877	0.973	0.973	0.973	0.974
V1	0.891	0.890	0.891	0.860	0.975	0.972	0.973	0.971
V2	0.921	0.918	0.919	0.891	0.981	0.981	0.981	0.974
V3	0.877	0.890	0.893	0.881	0.983	0.981	0.982	0.983
V5	0.898	0.888	0.894	0.886	0.980	0.978	0.979	0.977
V8	0.885	0.876	0.880	0.851	0.977	0.976	0.977	0.975
V9	0.907	0.893	0.902	0.886	0.980	0.978	0.980	0.977
V10	0.873	0.871	0.870	0.831	0.970	0.971	0.971	0.968
V13	0.866	0.886	0.889	0.875	0.982	0.981	0.982	0.982
V14	0.942	0.944	0.943	0.887	0.986	0.985	0.986	0.983
V17	0.930	0.936	0.936	0.878	0.984	0.982	0.983	0.976
p-value	0.6772	0.7674	4.8E-04		3.6E-05	1.1E-02	1.2E-05	

From the results reported in this chapter, it is observed that the RFCM is superior to other c -means algorithms. However, RFCM requires higher time compared to HCM. But, the performance of RFCM is significantly better than other c -means. Use of rough sets and fuzzy memberships adds a small computational load to HCM algorithm; however the corresponding integrated method (RFCM) shows a definite increase in Dunn and Silhouette and decrease in DB index.

The best performance of the RFCM algorithm is achieved due to the fact that the concept of crisp lower bound and fuzzy boundary of the RFCM algorithm deals with uncertainty, vagueness, and incompleteness in class definition; and membership function of the RFCM handles efficiently overlapping partitions. In effect, good cluster prototypes are obtained using

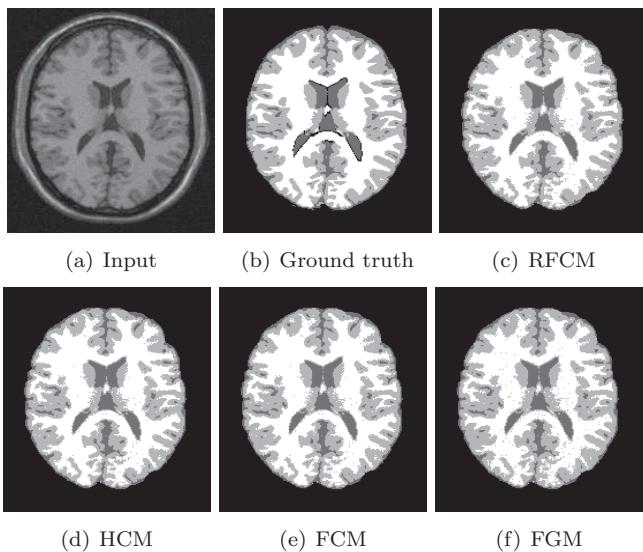


Fig. 15.5. Input image of BrainWeb with 20% bias and 5% noise, ground truth, and segmented images obtained using different algorithms.

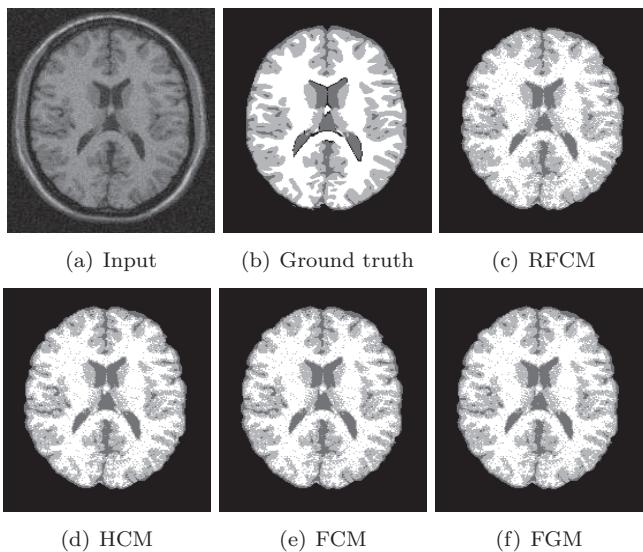


Fig. 15.6. Input image of BrainWeb with 20% bias and 9% noise, ground truth, and segmented images obtained using different algorithms.

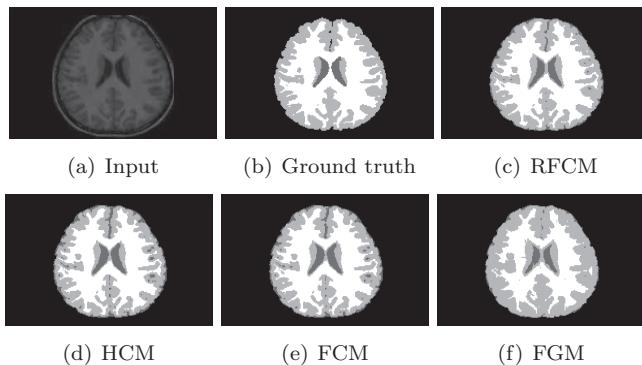


Fig. 15.7. Segmented images obtained by different algorithms for volume 14 of IBSR.

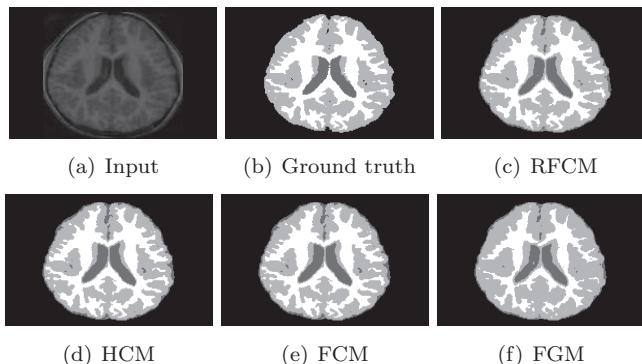


Fig. 15.8. Segmented images obtained by different algorithms for volume 17 of IBSR.

the RFCM algorithm with significantly lesser time. Although the methodology of integrating rough sets, fuzzy sets, and c -means algorithm has been efficiently demonstrated for clustering gene expression and microRNA expression data sets, and segmentation of brain MR images, the concept can be applied to other unsupervised classification problems.

References

- [1] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, NJ: Prentice Hall (1988).
- [2] A. K. Jain, M. N. Murty, and P. J. Flynn, Data Clustering: A Review, *ACM Computing Surveys*. **31**(3), 264–323 (1999).
- [3] J. McQueen. Some Methods for Classification and Analysis of Multivariate

- Observations. In *Proceedings of the 5th Berkeley Symposium on Mathematics, Statistics and Probability*, pp. 281–297 (1967).
- [4] R. E. Bellman, R. E. Kalaba, and L. A. Zadeh, Abstraction and Pattern Classification, *Journal of Mathematical Analysis and Applications*. **13**, 1–7 (1966).
 - [5] E. H. Ruspini, Numerical Methods for Fuzzy Clustering, *Information Sciences*. **2**, 319–350 (1970).
 - [6] J. C. Dunn, A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact, Well-Separated Clusters, *Journal of Cybernetics*. **3**, 32–57 (1974).
 - [7] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithm*. New York: Plenum (1981).
 - [8] R. Krishnapuram and J. M. Keller, A Possibilistic Approach to Clustering, *IEEE Transactions on Fuzzy Systems*. **1**(2), 98–110 (1993).
 - [9] R. Krishnapuram and J. M. Keller, The Possibilistic C-Means Algorithm: Insights and Recommendations, *IEEE Transactions on Fuzzy Systems*. **4**(3), 385–393 (1996).
 - [10] M. Barni, V. Cappellini, and A. Mecocci, Comments on “A Possibilistic Approach to Clustering”, *IEEE Transactions on Fuzzy Systems*. **4**(3), 393–396 (1996).
 - [11] F. Masulli and S. Rovetta, Soft Transition from Probabilistic to Possibilistic Fuzzy Clustering, *IEEE Transactions on Fuzzy Systems*. **14**(4), 516–527 (2006).
 - [12] N. R. Pal, K. Pal, J. M. Keller, and J. C. Bezdek, A Possibilistic Fuzzy C-Means Clustering Algorithm, *IEEE Transactions on Fuzzy Systems*. **13**(4), 517–530 (2005).
 - [13] H. Timm, C. Borgelt, C. Doring, and R. Kruse, An Extension to Possibilistic Fuzzy Cluster Analysis, *Fuzzy Sets and Systems*. **147**, 3–16 (2004).
 - [14] Z. Pawlak, *Rough Sets: Theoretical Aspects of Reasoning About Data*. Dordrecht, The Netherlands: Kluwer (1991).
 - [15] S. Hirano and S. Tsumoto, An Indiscernibility-Based Clustering Method with Iterative Refinement of Equivalence Relations: Rough Clustering, *Journal of Advanced Computational Intelligence and Intelligent Informatics*. **7**(2), 169–177 (2003).
 - [16] S. K. De, A Rough Set Theoretic Approach to Clustering, *Fundamenta Informaticae*. **62**(3–4), 409–417 (2004).
 - [17] P. Lingras and C. West, Interval Set Clustering of Web Users with Rough K-Means, *Journal of Intelligent Information Systems*. **23**(1), 5–16 (2004).
 - [18] S. K. Pal, B. D. Gupta, and P. Mitra, Rough Self Organizing Map, *Applied Intelligence*. **21**(3), 289–299 (2004).
 - [19] S. Ashraf, S. K. Shevade, and M. N. Murty, Rough Support Vector Clustering, *Pattern Recognition*. **38**, 1779–1783 (2005).
 - [20] P. Maji and S. K. Pal, *Rough-Fuzzy Pattern Recognition: Applications in Bioinformatics and Medical Imaging*. Wiley-IEEE Computer Society Press, New Jersey (2012).
 - [21] P. Maji and S. K. Pal, RFCM: A Hybrid Clustering Algorithm Using Rough

- and Fuzzy Sets, *Fundamenta Informaticae*. **80**(4), 475–496 (2007).
- [22] P. Maji and S. K. Pal, Rough Set Based Generalized Fuzzy C-Means Algorithm and Quantitative Indices, *IEEE Transactions on System, Man, and Cybernetics, Part B: Cybernetics*. **37**(6), 1529–1540 (2007).
 - [23] P. Maji and S. Paul, Robust Rough-Fuzzy C-Means Algorithm: Design and Applications in Coding and Non-Coding RNA Expression Data Clustering, *Fundamenta Informaticae*. **124**, 153–174 (2013).
 - [24] P. Maji and S. Paul, Rough-Fuzzy Clustering for Grouping Functionally Similar Genes from Microarray Data, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. **10**(2), 286–299 (2013).
 - [25] G. James, *Modern Engineering Mathematics*. Reading, MA: Addison-Wesley (1996).
 - [26] J. Bezdek, R. J. Hathaway, M. J. Sabin, and W. T. Tucker, Convergence Theory for Fuzzy C-Means: Counterexamples and Repairs, *IEEE Transactions on Systems, Man, and Cybernetics*. **17**, 873–877 (1987).
 - [27] H. Yan, Convergence Condition and Efficient Implementation of the Fuzzy Curve-Tracing (FCT) Algorithm, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*. **34**(1), 210–221 (2004).
 - [28] E. Domany, Cluster Analysis of Gene Expression Data, *Journal of Statistical Physics*. **110**, 1117–1139 (2003).
 - [29] J. Herrero, A. Valencia, and J. Dopazo, A Hierarchical Unsupervised Growing Neural Network for Clustering Gene Expression Patterns, *Bioinformatics*. **17**, 126–136 (2001).
 - [30] L. J. Heyer, S. Kruglyak, and S. Yoosheph, Exploring Expression Data: Identification and Analysis of Coexpressed Genes, *Genome Research*. **9**, 1106–1115 (1999).
 - [31] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. S. Lander, and T. R. Golub, Interpreting Patterns of Gene Expression with Self-Organizing Maps: Methods and Application to Hematopoietic Differentiation, *Proceedings of the National Academy of Sciences, USA*. **96**(6), 2907–2912 (1999).
 - [32] A. Ben-Dor, R. Shamir, and Z. Yakhini, Clustering Gene Expression Patterns, *Journal of Computational Biology*. **6**(3-4), 281–297 (1999).
 - [33] E. Hartuv and R. Shamir, A Clustering Algorithm Based on Graph Connectivity, *Information Processing Letters*. **76**(4-6), 175–181 (2000).
 - [34] R. Shamir and R. Sharan. CLICK: A Clustering Algorithm for Gene Expression Analysis. In *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology* (2000).
 - [35] E. P. Xing and R. M. Karp, CLIFF: Clustering of High-Dimensional Microarray Data via Iterative Feature Filtering Using Normalized Cuts, *Bioinformatics*. **17**(1), 306–315 (2001).
 - [36] C. Fraley and A. E. Raftery, How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis, *The Computer Journal*. **41**(8), 578–588 (1998).
 - [37] D. Ghosh and A. M. Chinnaiyan, Mixture Modelling of Gene Expression Data from Microarray Experiments, *Bioinformatics*. **18**, 275–286 (2002).

- [38] G. J. McLachlan, R. W. Bean, and D. Peel, A Mixture Model-Based Approach to the Clustering of Microarray Expression Data, *Bioinformatics*. **18**, 413–422 (2002).
- [39] K. Y. Yeung, C. Fraley, A. Murua, A. E. Raftery, and W. L. Ruzz, Model-Based Clustering and Data Transformations for Gene Expression Data, *Bioinformatics*. **17**, 977–987 (2001).
- [40] D. Jiang, J. Pei, and A. Zhang, DHC: A Density-Based Hierarchical Clustering Method for Time-Series Gene Expression Data. In *Proceedings of the 3rd IEEE International Symposium on Bioinformatics and Bioengineering* (2003).
- [41] D. Jiang, C. Tang, and A. Zhang, Cluster Analysis for Gene Expression Data: A Survey, *IEEE Transactions on Knowledge and Data Engineering*. **16**(11), 1370–1386 (2004).
- [42] A. Brazma and J. Vilo, Minireview: Gene Expression Data Analysis, *Federation of European Biochemical Societies*. **480**, 17–24 (2000).
- [43] P. Maji and S. Paul, *Scalable Pattern Recognition Algorithms: Applications in Computational Biology and Bioinformatics*, p. 304. Springer-Verlag, London (April, 2014). ISBN 978-3-319-05629-6.
- [44] J. P. Rousseeuw, Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis, *Journal of Computational and Applied Mathematics*. **20**, 53–65 (1987).
- [45] J. C. Bezdek and N. R. Pal, Some New Indexes for Cluster Validity, *IEEE Transactions on System, Man, and Cybernetics, Part B: Cybernetics*. **28**, 301–315 (1988).
- [46] D. Dembele and P. Kastner, Fuzzy C-Means Method for Clustering Microarray Data, *Bioinformatics*. **19**(8), 973–980 (2003).
- [47] E. I. Boyle, S. Weng, J. Gollub, H. Jin, D. Botstein, J. M. Cherry, and G. Sherlock, GO::Term Finder Open Source Software for Accessing Gene Ontology Information and Finding Significantly Enriched Gene Ontology Terms Associated with a List of Genes, *Bioinformatics*. **20**, 3710–3715 (2004).
- [48] J. L. Sevilla, V. Segura, A. Podhorski, E. Guruceaga, J. M. Mato, L. A. Martinez-Cruz, F. J. Corrales, and A. Rubio, Correlation Between Gene Expression and GO Semantic Similarity, *IEEE/ACM Transactions on Computational Biology and Bioinformatics*. **2**(4), 330–338 (2005).
- [49] C. Wang, S. Yang, G. Sun, X. Tang, S. Lu, O. Neyrolles, and Q. Gao, Comparative miRNA Expression Profiles in Individuals with Latent and Active Tuberculosis, *PLoS One*. **6**(10) (2011).
- [50] J. Pevsner, *Bioinformatics and Functional Genomics*. Wiley-Blackwell (2009).
- [51] C. G. Liu, G. A. Calin, S. Volinia, and C. M. Croce, MicroRNA Expression Profiling Using Microarrays, *Nature Protocols*. **3**(4), 563–578 (2008).
- [52] V. Ambros, Control of Developmental Timing in *Caenorhabditis Elegans*, *Current Opinion in Genetics and Development*. **10**(4), 428–433 (2000).
- [53] Y. Grad, J. Aach, G. D. H., B. J. Reinhart, G. M. Church, G. Ruvkun, and J. Kim, Computational and Experimental Identification of *C. elegans* microRNAs, *Molecular Cell*. **11**(5), 1253–1263 (2003).

- [54] B. John, A. J. Enright, A. Aravin, T. Tuschl, C. Sander, and S. M. Debora, Human MicroRNA Targets, *PLoS Biology*. **2**(11) (2004).
- [55] E. Enerly, I. Steinfeld, K. Kleivi, S. K. Leivonen, M. R. Aure, H. G. Russnes, J. A. Ronneberg, H. Johnsen, R. Navon, E. Rodland, R. Makela, B. Naume, M. Perala, O. Kallioniemi, V. N. Kristensen, Z. Yakhini, and A. L. B. Dale, miRNA-mRNA Integrated Analysis Reveals Roles for miRNAs in Primary Breast Tumors, *PLoS One*. **6**(2) (2011).
- [56] J. Lu, G. Getz, E. A. Miska, E. A. Saavedra, J. Lamb, D. Peck, A. S. Cordero, B. L. Ebert, R. H. Mak, A. A. Ferrando, J. R. Downing, T. Jacks, H. R. Horvitz, and T. R. Golub, MicroRNA Expression Profiles Classify Human Cancers, *Nature Letters*. **435**(9), 834–838 (2005).
- [57] S. Paul and P. Maji, City Block Distance and Rough-Fuzzy Clustering for Identification of Co-Expressed microRNAs, *Molecular BioSystems*. **10**(6), 1509–1523 (2014).
- [58] J. C. Bezdek, L. O. Hall, and L. P. Clarke, Review of MR Image Segmentation Techniques Using Pattern Recognition, *Medical Physics*. **20**(4), 1033–1048 (1993).
- [59] P. Maji, M. K. Kundu, and B. Chanda, Second Order Fuzzy Measure and Weighted Co-Occurrence Matrix for Segmentation of Brain MR Images, *Fundamenta Informaticae*. **88**(1-2), 161–176 (2008).
- [60] P. K. Sahoo, S. Soltani, A. K. C. Wong, and Y. C. Chen, A Survey of Thresholding Techniques, *Computer Vision, Graphics, and Image Processing*. **41**, 233–260 (1988).
- [61] I. N. Manousakes, P. E. Undrill, and G. G. Cameron, Split and Merge Segmentation of Magnetic Resonance Medical Images: Performance Evaluation and Extension to Three Dimensions, *Computers and Biomedical Research*. **31**(6), 393–412 (1998).
- [62] J. F. Mangin, V. Frouin, I. Bloch, J. Rgis, and J. Lpez-Krahe, From 3D Magnetic Resonance Images to Structural Representations of the Cortex Topography Using Topology Preserving Deformations, *Journal of Mathematical Imaging and Vision*. **5**(4), 297–318 (1995).
- [63] S. Cagnoni, G. Coppini, M. Rucci, D. Caramella, and G. Valli, Neural Network Segmentation of Magnetic Resonance Spin Echo Images of the Brain, *Journal of Biomedical Engineering*. **15**(5), 355–362 (1993).
- [64] L. O. Hall, A. M. Bensaid, L. P. Clarke, R. P. Velthuizen, M. S. Silbiger, and J. C. Bezdek, A Comparison of Neural Network and Fuzzy Clustering Techniques in Segmenting Magnetic Resonance Images of the Brain, *IEEE Transactions on Neural Network*. **3**(5), 672–682 (1992).
- [65] W. E. Reddick, J. O. Glass, E. N. Cook, T. D. Elkin, and R. J. Deaton, Automated Segmentation and Classification of Multispectral Magnetic Resonance Images of Brain Using Artificial Neural Networks, *IEEE Transactions on Medical Imaging*. **16**(6), 911–918 (1997).
- [66] T. McInerney and D. Terzopoulos, T-Snakes: Topology Adaptive Snakes, *Medical Image Analysis*. **4**(2), 73–91 (2000).
- [67] N. C. Andreasen, R. Rajarethinam, T. Cizadlo, S. Arndt, V. W. Swayze, L. A. Flashman, D. S. O'Leary, J. C. Ehrhardt, and W. T. Yuh, Automatic

- Atlas-Based Volume Estimation of Human Brain Regions from MR Images, *Journal of Computer Assisted Tomography*. **20**(1), 98–106 (1996).
- [68] G. E. Christensen, S. C. Joshi, and M. I. Miller, Volumetric Transformation of Brain Anatomy, *IEEE Transactions on Medical Imaging*. **16**(6), 864–877 (1997).
 - [69] Y. Li and Z. Chi, MR Brain Image Segmentation Based on Self-Organizing Map Network, *International Journal of Information Technology*. **11**(8), 45–53 (2005).
 - [70] M. G. Bello, A Combined Markov Random Field and Wave-Packet Transform-Based Approach for Image Segmentation, *IEEE Transactions on Image Processing*. **3**(6), 834–846 (1994).
 - [71] P. Maji and S. Roy, Rough-Fuzzy Clustering and Unsupervised Feature Selection for Wavelet Based MR Image Segmentation, *PLoS ONE*. **10**(4), e0123677 (2015). doi: 10.1371/journal.pone.0123677.
 - [72] G. N. Abras and V. L. Ballarin, A Weighted K-Means Algorithm Applied to Brain Tissue Classification, *Journal of Computer Science and Technology*. **5**(3), 121–126 (2005).
 - [73] B. Vemuri, S. Rahman, and J. Li, Multiresolution Adaptive K-Means Algorithm for Segmentation of Brain MRI, *Image Analysis Applications and Computer Graphics*. **1024**, 347–354 (1995).
 - [74] X. Chen, J. K. Udupa, U. Bagci, Y. Zhuge, and J. Yao, Medical Image Segmentation by Combining Graph Cuts and Oriented Active Appearance Models, *IEEE Transactions on Image Processing*. **21**(4), 2035–2046 (2012).
 - [75] M. E. Brandt, T. P. Bohan, L. A. Kramer, and J. M. Fletcher, Estimation of CSF, White and Gray Matter Volumes in Hydrocephalic Children Using Fuzzy Clustering of MR Images, *Computerized Medical Imaging and Graphics*. **18**, 25–34 (1994).
 - [76] C. L. Li, D. B. Goldgof, and L. O. Hall, Knowledge-Based Classification and Tissue Labeling of MR Images of Human Brain, *IEEE Transactions on Medical Imaging*. **12**(4), 740–750 (1993).
 - [77] K. Xiao, S. H. Ho, and A. E. Hassanien, Automatic Unsupervised Segmentation Methods for MRI Based on Modified Fuzzy C-Means, *Fundamenta Informaticae*. **87**(3-4), 465–481 (2008).
 - [78] D. L. Pham and J. L. Prince, Adaptive Fuzzy Segmentation of Magnetic Resonance Images, *IEEE Transactions on Medical Imaging*. **18**(9), 737–752 (1999).
 - [79] P. Maji and S. K. Pal, Maximum Class Separability for Rough-Fuzzy C-Means Based Brain MR Image Segmentation, *LNCS Transactions on Rough Sets*. **9**, 114–134 (2008).
 - [80] W. M. Wells, III, W. E. L. Grimson, R. Kikins, and F. A. Jolezs, Adaptive Segmentation of MRI Data, *IEEE Transactions on Medical Imaging*. **15**(8), 429–442 (1996).
 - [81] Z. Liang, J. R. MacFall, and D. P. Harrington, Parameter Estimation and Tissue Segmentation from Multispectral MR Images, *IEEE Transactions on Medical Imaging*. **13**(3), 441–449 (1994).
 - [82] H. Greenspan, A. Ruf, and J. Goldberger, Constrained Gaussian Mixture

- Model Framework for Automatic Segmentation of MR Brain Images, *IEEE Transactions on Medical Imaging*. **25**(9), 1233–1245 (2006).
- [83] S. M. Smith, Fast Robust Automated Brain Extraction, *Human Brain Mapping*. **17**(3), 143–155 (2002).
- [84] A. Banerjee and P. Maji, Rough Sets for Bias Field Correction in MR Images Using Contraharmonic Mean and Quantitative Index, *IEEE Transactions on Medical Imaging*. **32**(11), 2140–2151 (2013).
- [85] N. Otsu, A Threshold Selection Method from Gray Level Histogram, *IEEE Transactions on System, Man, and Cybernetics*. **9**(1), 62–66 (1979).

Chapter 16

Keygraphs: Structured Features for Object Detection and Applications

M. Hashimoto¹, H. Morimitsu², R. Hirata-Jr.² and R. M. Cesar-Jr.²

¹*Inspur - Instituto de Ensino e Pesquisa
São Paulo, Brazil*

marcelo.hashimoto@insper.edu.br

²*e-Science Laboratory, Department of Computer Science
University of São Paulo, São Paulo, Brazil
{henriquem87, hirata, cesar}@vision.ime.usp.br*

Object detection is one of the most important problems in computer vision and it is the base for many others, such as navigation, stereo matching and augmented reality. One of the most popular and powerful choices for performing object detection is using keypoint correspondence approaches. Several keypoint detectors and descriptors have already been proposed but they often extract information from the neighborhood of each point individually, without considering the structure and relationship between them. Exploring structural pattern recognition techniques is a powerful way to fill this gap. In this chapter the concept of keygraphs is explored for extracting structural features from regular keypoints. Keygraphs provide more flexibility to the description process and are more robust than traditional keypoint descriptors, such as SIFT and SURF, because they rely on structural information. The results observed in different tests show that this simplicity significantly improves the time performance, while also keeping them highly discriminative. The effectiveness of keygraphs is validated by using them to detect objects in real-time applications on a mobile phone.

16.1. Introduction

Object detection is a well-known problem in computer vision that consists in: given a model representing an object and a scene, decide if there are instances of the object in the scene and, if so, estimate the pose of each instance (see Fig. 16.1). Solving this problem is useful for many applica-

tions like automatic surveillance, medical analysis, and augmented reality, but particularly challenging due to difficulties like image distortions, background clutter, and partial occlusions.

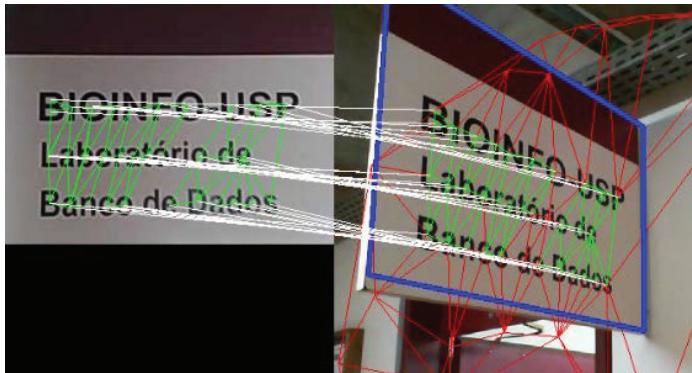


Fig. 16.1. Example of object detection.

Among the techniques for object detection available in the literature, methods based on keypoint correspondence have been particularly successful [1–5]. The outline of such methods consists in: (1) detecting interest points in the model and the scene, (2) selecting correspondences between the two sets of points by analyzing the similarity between photometric descriptors, and (3) applying a fitting procedure over those correspondences. This framework is not specific to object detection and has also been used for robot navigation [6], feature tracking [7–9], stereo matching [7, 10, 11], image retrieval [12], 3D reconstruction [4] and visual odometry [13].

In this chapter, a framework for object detection using structured features is described. This method is based on replacing keypoints by keygraphs, i.e. isomorph directed graphs whose vertices are keypoints, in the aforementioned outline. The motivation for this proposal is exploring relative and structural information that an individual point cannot provide. Because keygraphs can be more numerous and complex, such replacement is not straightforward. Two issues must be considered: a possible loss of locality, since large structures might be built, and a possible combinatorial explosion, since large sets might be enumerated. Therefore, in addition to describing the framework itself, more details about the implementation to take those issues into account are explained using some practical applications.

16.2. Basic concepts

Some basic definitions necessary for the complete understanding of the described methods are presented here. The reader interested in more details is referred to [14–16].

A **graph** is a pair $G = (\mathcal{V}, \mathcal{E})$, where \mathcal{V} is a finite set of vertices, or nodes, and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a finite set of edges. If G is a directed graph, then \mathcal{E} is a set of ordered pairs (also called directed edges or arcs), i.e., $(u, v) \neq (v, u)$.

An **isomorphism** between two graphs, $G = (\mathcal{V}_G, \mathcal{E}_G)$ and $H = (\mathcal{V}_H, \mathcal{E}_H)$, is a bijective function $f : \mathcal{V}_G \rightarrow \mathcal{V}_H$ with edge preservation, i.e. an edge $(u, v) \in \mathcal{E}_G$ if and only if $(f(u), f(v)) \in \mathcal{E}_H$.

When using graphs to represent images, it is common to associate the vertices to the more salient and relevant features of the image and then to use the edges to express their spatial relation. For example, for human body pose estimation, a common approach is to first separate the body into several parts (e.g., arms, legs, etc.) and then to extract information from each of them. In this case, each of those parts can be considered as one feature and a graph can be built by associating a vertex with each part and by adding edges to obtain a significant spatial relation between them, e.g., representing the human skeleton [17, 18]. When dealing with objects in general, it is often difficult to find such well-defined important points to assign vertices. Therefore, a less specialized approach is required. Practical examples include choosing points where the image gradient change abruptly or, conversely, the centroid of regions where they do not change.

Regardless of how they are obtained, in computer vision such relevant points are commonly referred as features, interest points or **keypoints**. Along this chapter, the latter term will be used for simplicity and to avoid confusion with the concept of feature vectors, which is also present in this work. More formally, they are obtained as a subset $\mathcal{K} \subseteq \mathcal{P}$ of all pixels \mathcal{P} of the image.

The method implemented to find such keypoints is called a **keypoint detector**. Several different detectors exist that define specific heuristics about what is considered a keypoint and how to find it. Most of them, including the most used detectors such as SIFT [2] and SURF [4], are based on corner detection. In other words, these detectors search for points that belong to corners of objects and choose them as the keypoints. Although widely used, corners are not the only option. Another interesting approach, for example, is based on finding homogeneous regions and choosing their

centroids as keypoints. Examples of such detectors are MSER [10] and MFD [19].

Besides finding the keypoints, it is also important to make them comparable, i.e. it is necessary to describe each point so that it is possible to determine the similarities among them. The information associated with each keypoint is called a **descriptor**. Since they are often used to describe the appearance of the keypoint or the image, they are referred to as image or **appearance descriptors**. An appearance descriptor can be defined as a vector $D \in R^n$ given by the transformation $d : f(k) \rightarrow D$, where $k \subseteq \mathcal{K}$ is one keypoint and $f(k)$ is a function that gives a set of points related to k , usually a set of neighboring points. The transformation d may be defined in different ways, but it may consider the values of the pixels in \mathcal{Q} , their positions and global gradient orientation, among other characteristics.

Several descriptors are available and they can be separated in two categories: global and local descriptors. Global descriptors allow the whole image to be described by a single vector. Because the comparison of images can be reduced to the comparison of a single descriptor, global descriptors are commonly adopted for exact image retrieval, where a query image is used to find similar ones, usually in a large database. Examples of global descriptors include GIST [20], color histograms, and image mean, among others.

Keypoints are described using local descriptors. These descriptors, as the name implies, only describe a small region of the image, and hence multiple local descriptors are used in a single image. Local descriptors are more suitable for object recognition because the local property of the descriptor allows detecting them independently of each other. Such a characteristic makes object detection much more robust, because even when some points are occluded or change significantly, the others can still be used to obtain a good result. That makes it possible to find objects even under adverse conditions such as occlusion or perspective transformations [21].

The descriptors are usually obtained by extracting information from the pixels of the neighborhood of each point. Depending on the image and the object, the neighborhood size should be carefully considered. Some detectors (SIFT, SURF) calculate precise regions as part of the detection process, but others (Harris) do not. Furthermore, more robust descriptors also usually assign an orientation to each patch, making the whole process computationally intense.

Graphs can be useful for providing more information and flexibility by taking advantage of their structural properties. As the arcs between points

provide information about the neighborhood of each point, it is possible to avoid having to explicitly compute the optimal patch size and orientation by relying on the size and orientation of the arcs. Moreover, other kinds of descriptors that are not only restricted to areas around the keypoint can also be extracted.

16.3. Overview of the keygraph framework

Keygraphs are a generalization of regular keypoints in the sense that keypoints are used as vertices of the graphs and then arcs are added to connect them. In other words, a **keygraph** is a graph defined by the pair $(\mathcal{V}, \mathcal{A})$ where $\mathcal{V} \subseteq \mathcal{P}$ is a set of keypoints, $\mathcal{A} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of arcs. To deal with the issue of comparability, keygraphs are constrained so that for every pair of keygraphs, there must be an isomorphism between them. Unlike a pair of points, a pair of graphs can have structural differences and therefore requires structural criteria for deciding whether it is comparable or not. In that sense, the bijection can be used as a criterion to purposefully avoid mapping multiple model keypoints to a single scene keypoint. This contrasts, for example, with works where super-segmentation graphs are mapped to segmentation graphs [22]: in such works, surjections are more adequate. The criterion of arc preservation, in its turn, avoids mapping an arc to a pair that is not an arc. This allows part-wise analysis: the isomorphism establishes which model vertex should be compared to each scene vertex and which model arc should be compared to each scene arc.

Note that the keygraph framework is a generalization of the keypoint framework: a keypoint can be interpreted as a keygraph with a single vertex, and from a single-vertex graph to another there is always an isomorphism. A desired property is that keygraphs should provide local descriptors of the image, keeping the detection robustness shown by the independence of regular local keypoints.

Table 16.1 illustrates the analogy between the existing keypoint framework and the proposed keygraph framework.

The numbering represents the four main steps that both frameworks analogously have: detection, description, correspondence, and pose estimation. Keypoint detection is the first task of both frameworks, and the only task that does not require adaptation for the keygraph framework: existing algorithms can be used directly.

Figure 16.2 illustrates the sequence of steps defined by the proposed framework.

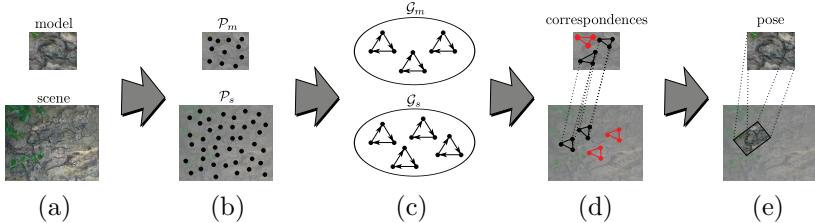


Fig. 16.2. Illustration of the framework. (a) Model and scene. (b) Detection of keypoints \mathcal{K}_m and \mathcal{K}_s . (c) Detection of keygraphs \mathcal{G}_m and \mathcal{G}_s . (d) Selection of correspondences. (e) Pose estimation.

Firstly, independent sets of model keygraphs \mathcal{G}_m and scene keygraphs \mathcal{G}_s are detected on the respective images (Fig. 16.2(c)). Afterwards, the sets of keygraphs are used to extract descriptors from each image. In the keygraph framework, a descriptor is not global with respect to the graph: the descriptors are computed locally and the correspondences are selected by part-wise analysis. Then, the described keygraphs are compared to find correspondences between both images (Fig. 16.2(d)). Finally, a pose can be estimated using these correspondences (Fig. 16.2(e)). In the following sections, all the aforementioned steps will be further explained.

16.3.1. Keygraph detection

Keygraphs are detected by taking a set of keypoints as its vertices and by adding arcs between them. In theory, keygraphs can be built over any set of keypoints. However, empirical tests showed that some keypoint detectors provide more suitable points for keygraph building. According to the tests, the Hessian-Affine [23, 24] and MSER [10] showed better results on the

Table 16.1. Analogy between the existing keypoint framework and the proposed keygraph framework.

keypoint framework	keygraph framework
1) detect model keypoints \mathcal{K}_m and scene keypoints \mathcal{K}_s	1) detect model keypoints \mathcal{K}_m and scene keypoints \mathcal{K}_s
2) compute the descriptor of each keypoint	detect model keygraphs \mathcal{G}_m and scene keygraphs \mathcal{G}_s
3) select keypoint correspondences	2) compute the descriptors of each keygraph
4) estimate pose using the selected correspondences	3) select keygraph correspondences 4) estimate pose using the selected correspondences

database shared by Mikolajczyk and Schmid [25] and some other databases. The first detector is based on extrema of Hessian determinants over multiple scales and the second is based on extremal regions that are maximally stable. Both are actually covariant region detectors, converted to keypoint detectors by taking the region centroids. The reason why such keypoints are more suitable for keygraphs will be clearer after understanding how its descriptors are extracted (Sec. 16.3.2).

In order to keep the good properties of the keypoints, it is interesting to create keygraphs that keep the constraint of locality. In other words, keygraphs should not cover large portions of the image. In order to be able to compare them efficiently, it is also interesting to keep them as simple as possible. Section 16.3.5 elaborates more on how keygraph building can be restricted to keep the desired properties.

16.3.2. Descriptor extraction

The descriptor chosen for the arcs is based on Fourier coefficients of an intensity profile and it is inspired by Tell and Carlsson [26, 27]. An intensity profile is simply a sequence of the values of the pixels that are under the corresponding arc. The Bresenham algorithm [28] may be adopted to obtain the values of the profile. Given an intensity profile $f: \{0, \dots, l-1\} \rightarrow \{0, \dots, 255\}$ (see Fig. 16.3), its discrete Fourier transform is the function $F: \{0, \dots, l-1\} \rightarrow \mathbb{C}$ defined by

$$F(x) = \sum_{y=0}^{l-1} e^{-\frac{2\pi i}{l} xy} f(y)$$

and its descriptor is defined by

$$(a_0, b_0, \dots, a_{m-1}, b_{m-1})$$

where a_x and b_x are, respectively, the real and imaginary parts of $F(x)$.

Choosing only the lowest-frequency coefficients ensures both a shorter descriptor size and a greater robustness to noise.

Intensity profiles are naturally invariant to rotation. They are not invariant to perspective transforms, but are sufficiently robust under the maximum distance criterion shown in Section 16.3.5, since small local regions are not so affected by the perspective transform. Discarding $F(0)$ ensures invariance to brightness and normalizing the descriptor vector by

$$\sqrt{\sum_{x=1}^m (a_x^2 + b_x^2)}$$

ensures invariance to contrast.

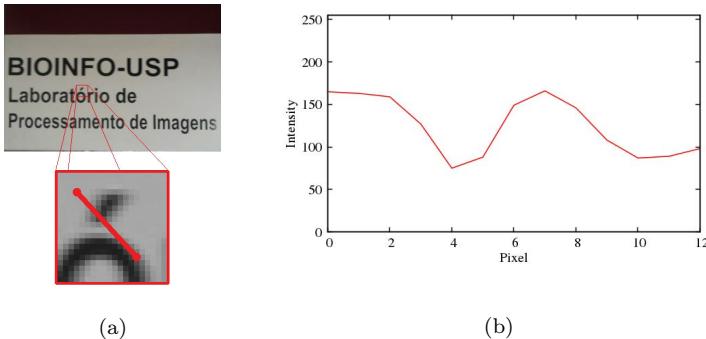


Fig. 16.3. (a) an object and a possible arc. (b) the intensity profile of that arc.

Profile-based descriptors are inherently poorer than region-based descriptors, but also have inherent advantages.

- **Fast computation.** SIFT is widely recognized as one of the most accurate descriptors in the literature, but it is also known for its slow computation. The focus of many recent works [4, 11, 13, 29] is keeping the accuracy while reducing running time. This makes the simplicity of intensity profiles particularly interesting.
- **Low dimensionality.** Empirical results have shown that good descriptors can be obtained using only 3 Fourier coefficients. On the other hand, besides the slow computation, SIFT also has a high dimensionality (128). This raises the difficulty of building indexing structures due to the curse of dimensionality, i.e. the exponential increase in volume that reduces the significance of dissimilarity functions. In fact, SIFT descriptors have been used as experimental data for indexing structure proposals [30, 31] and dimensionality reduction procedures have also been used in the literature [25, 32]. This makes the simplicity of intensity profile further interesting.
- **Natural robustness.** Since regions are not naturally invariant to rotation like intensity profiles, invariance must be provided by the descriptor. SIFT and SURF, for example, assign an orientation to each keypoint and adapt accordingly. Furthermore, intensity profiles provide a well-defined concept of size for normalization: the distance between the two vertices of the respective arc. In contrast, not all keypoint detectors provide such concept: although there are detectors that provide scale values [1, 2, 4], elliptical regions [23, 24], and complete contours [10], there are also detectors

that do not provide anything [33].

- **Weaker discrimination.** Individually, regions are more discriminative than profiles. However, this apparent advantage becomes a shortcoming in part-wise analysis: if descriptors are designed to be as discriminative as possible individually, they can be excessively discriminative collectively. For example: using the distance between SIFT descriptors to decide whether two single keypoints are similar has a high chance of success, but using the same criterion to compare two sets of keypoints might be unreasonably harsh.

For additional robustness, three smoothing procedures are applied over the intensity profiles before computing the Fourier transform. First, a Gaussian blur is applied to the image. Then, the mean of parallel intensity profiles is computed, as illustrated by Fig. 16.4. Finally, this mean is scaled to a length l with linear interpolation. Empirical results showed that the best results were obtained by setting $l = 30$.

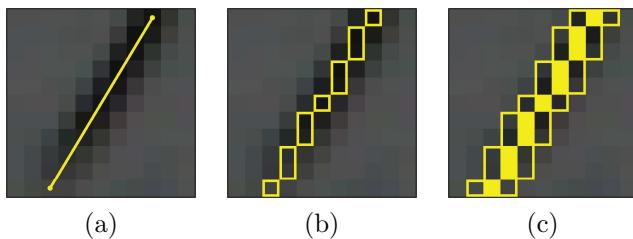


Fig. 16.4. Mean of intensity profiles. (a) The straight line between the two vertices of an arc. (b) The sequence of pixels intersected by the line. (c) The same pixels and one additional parallel sequence on each side: the Fourier transform is computed from a scaled mean of the three sequences.

As the descriptors are extracted from the arcs of the keygraphs, it is usually better that the vertices (keypoints) do not lie on the borders of the image. The reason is that, if they are on the borders, many times the arcs will also be over the borders, creating very poor intensity profiles (see Fig. 16.5). That explains why the choice of region keypoint detectors, such as MSER and Hessian-Affine, are preferred over corner detectors like SIFT and Harris.

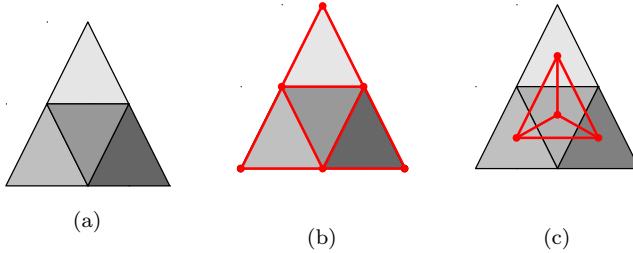


Fig. 16.5. Suppose (a) is the object to be detected. If a corner keypoint detector is used, all the arcs (in red) will be on the borders of the object (b) and hence, the keygraph descriptors will be poor. Centroid detectors yield more interesting arcs (c), which provide better descriptors.

16.3.3. Keygraph correspondence

A keygraph correspondence is a triple (G_m, G_s, ι) , where G_m is a model keygraph, G_s is a scene keygraph, and ι is an isomorphism from G_m to G_s . Specifying the isomorphism is the solution proposed for the issue of multi-comparability: from one graph to another, more than one isomorphism can exist. Figure 16.6 shows an example.

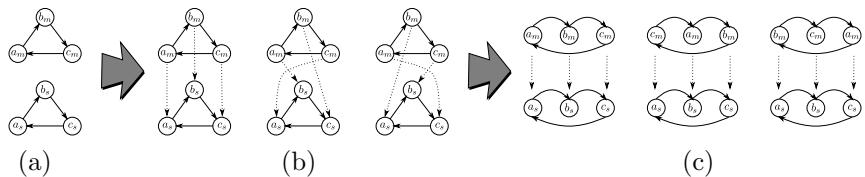


Fig. 16.6. Example of multi-comparability. (a) A model keygraph with vertices a_m, b_m, c_m and a scene keygraph with vertices a_s, b_s, c_s . (b) Visual representation of the three isomorphisms from the model keygraph to the scene keygraph: for each isomorphism ι , a dashed arrow from a model vertex v to a scene vertex w illustrates that $\iota(v) = w$. (c) The same representation, after repositioning the vertices to emphasize arc preservation.

In contrast, a keypoint correspondence is simply a pair (p_m, p_s) , where p_m is a model keypoint and p_s is a scene keypoint. Note that a keygraph correspondence (G_m, G_s, ι) implies the set of keypoint correspondences

$$\{(v, \iota(v)) : v \text{ is a vertex of } G_m\} \quad (16.1)$$

and therefore a set of keygraph correspondences Γ implies the set of key-

point correspondences

$$\Pi := \bigcup_{(G_m, G_s, \iota) \in \Gamma} \{(v, \iota(v)): v \text{ is a vertex of } G_m\} \quad (16.2)$$

The first implication (Eq. 16.1) is particularly relevant for estimating pose using RANSAC, as shown in Section 16.3.3.1, and the second implication is particularly relevant for analyzing the experimental results.

It is assumed that there are few models but many scenes: this is reasonable for applications such as video tracking and content-based image retrieval. Under this assumption, detecting model keygraphs and computing their respective descriptors once and storing them for subsequent usage is more adequate than repeating such detection and computation for each scene. Therefore, an asymmetric version of the framework is implemented (Table 16.2).

Table 16.2. Algorithm for keygraphs correspondence.

1	detect and store model keypoints \mathcal{K}_m
2	detect and store model keygraphs \mathcal{G}_m
3	for each G_m in \mathcal{G}_m
4	compute and store the descriptors of G_m
5	for each scene
6	detect scene keypoints \mathcal{K}_s
7	detect scene keygraphs \mathcal{G}_s
8	for each G_s in \mathcal{G}_s
9	compute the descriptors of G_s
10	for each G_m in \mathcal{G}_m
11	for each isomorphism ι from G_m to G_s
12	select or reject (G_m, G_s, ι)
13	estimate pose using the selected correspondences

It is also assumed that there is enough time available for this storage: only correspondence selection and pose estimation must be as fast as possible. With this assumption the model keygraphs and their respective descriptors can be processed beforehand to make the correspondence selection faster. This paradigm has been successful in the literature of keypoint correspondence, particularly in works based on indexing structures [1, 2] and machine learning [3, 34].

16.3.3.1. Keytuples

Given a set of model keygraphs and a set of scene keygraphs, obtaining correspondences is not straightforward because a bijection from a model graph vertex set to a scene graph vertex set might not be an isomorphism. In Fig. 16.6, for example, the three isomorphisms represent only half of the six possible bijections. This raises an issue:

- on one hand, the exhaustive verification of all bijections is expensive: if the keygraphs have k vertices, there are $k!$ possible bijections;
- on the other hand, an isomorphism to a scene graph cannot be stored when only model graphs are available.

In order to solve this issue, the concept of **keytuples** is introduced. Let $G = (\mathcal{V}, \mathcal{A})$ be a keygraph with k vertices, (v_1, \dots, v_k) be an ordering of \mathcal{V} , and σ be a $k \times k$ binary matrix (adjacency matrix). The tuple (v_1, \dots, v_k) is called a keytuple of G with respect to σ if

$$\sigma_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \text{ belongs to } \mathcal{A}; \\ 0 & \text{otherwise.} \end{cases}$$

In other words, σ establishes for each i, j whether there is an arc in G from the i -th to the j -th vertex of the keytuple. In Fig. 16.6, for example, the orderings $(a_m, b_m, c_m), (c_m, a_m, b_m), (b_m, c_m, a_m)$ are keytuples of the model graph with respect to

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

because this matrix establishes that there are arcs from the first to the second vertex, from the second to the third vertex, and from the third to the first vertex.

From the definition of isomorphism and the definition of keytuple, it is easy to prove the following proposition.

Proposition 16.1. *Let $G_m = (\mathcal{V}_m, \mathcal{A}_m), G_s = (\mathcal{V}_s, \mathcal{A}_s)$ be keygraphs, σ be a matrix, (w_1, \dots, w_k) be a keytuple of G_s with respect to σ , and \mathcal{T} be the set of keytuples of G_m with respect to σ . Then a bijection $\iota: \mathcal{V}_m \rightarrow \mathcal{V}_s$ is an isomorphism from G_m to G_s if and only if there exists a keytuple (v_1, \dots, v_k) in \mathcal{T} such that $\iota(v_i) = w_i$ for all i .*

This allows a straightforward procedure for obtaining all isomorphisms from a model keygraph G_m to a scene keygraph G_s given a matrix σ :

- (1) let \mathcal{T}_m be the set of all keytuples of G_m with respect to σ ;
- (2) let (w_1, \dots, w_k) be a keytuple of G_s with respect to σ ;
- (3) for each keytuple (v_1, \dots, v_k) in \mathcal{T}_m , let ι be the isomorphism from G_m to G_s such that $\iota(v_i) = w_i$ for all i .

Such procedure fits adequately into the paradigm of prioritizing exhaustiveness when extracting model data and prioritizing speed when extracting scene data: Step 1 considers all model keytuples, while Step 2 chooses a single scene keytuple. The proposition ensures the choice can be arbitrary.

Figure 16.7 shows an example of keytuples and the following outline adapts the asymmetric framework to them.

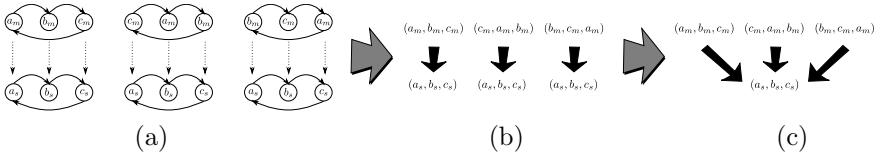


Fig. 16.7. Continuation of Fig. 16.6 for an example of keytuples. The dashed arrows in illustration (a) are not necessary because the isomorphisms are implied by vertex ordering. Illustration (b) is simpler but redundant because the same scene keytuple is repeated. Illustration (c) emphasizes the asymmetry.

Note that this adaptation replaced a double loop in keygraphs and isomorphisms by a single loop in keytuples (lines 10-12 in Table 16.2 for lines 13-16 in Table 16.3).

Table 16.3. Algorithm using keytuples.

1	detect and store model keypoints \mathcal{K}_m
2	detect and store model keygraphs \mathcal{G}_m
3	for each G_m in \mathcal{G}_m
4	compute and store the descriptors of G_m
5	for each keytuple (v_1, \dots, v_k) of G_m
6	store (v_1, \dots, v_k) in \mathcal{T}_m
7	for each scene
8	detect scene keypoints \mathcal{K}_s
9	detect scene keygraphs \mathcal{G}_s
10	for each G_s in \mathcal{G}_s
11	compute the descriptors of G_s
12	let (w_1, \dots, w_k) be a keytuple of G_s
13	for each keytuple (v_1, \dots, v_k) in \mathcal{T}_m
14	let G_m be the graph in \mathcal{G}_m such that (v_1, \dots, v_k) is a keytuple of G_m
15	let ι be the isomorphism from G_m to G_s such that $\iota(v_i) = w_i$ for all i
16	select or reject (G_m, G_s, ι)
17	estimate pose using the selected correspondences

The procedure for selecting or rejecting a keygraph correspondence is a simple thresholding. For each arc correspondence implied by the isomorphism, a dissimilarity function is applied to the descriptors, and the correspondence is rejected if one of the dissimilarities is higher than a certain threshold, as illustrated by Fig. 16.8.

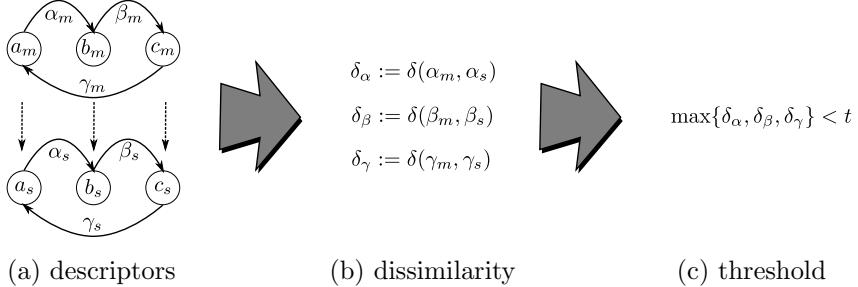


Fig. 16.8. Example of keygraph correspondence selection using arc dissimilarity thresholding. (a) An isomorphism from a model keygraph with arc descriptors $\alpha_m, \beta_m, \gamma_m$ to a scene keygraph with arc descriptors $\alpha_s, \beta_s, \gamma_s$. (b) The three arc dissimilarities implied by the isomorphism. (c) The thresholding condition for selecting a correspondence.

16.3.4. Pose estimation

A RANSAC-based pose estimation step is responsible for recognizing the correspondences that have been incorrectly selected and ignoring them. RANSAC is a probabilistic fitting procedure, specifically designed to handle high proportions of outliers. In general terms, it is based on iteratively choosing random sets of correspondences: for each set, a pose candidate is estimated and the number of correspondences that are inliers with respect to such candidate is stored as its score. The number of iterations is probabilistically adapted according to the best score: a high number of inliers implies a lower expected number of iterations before finding an optimal candidate. The actual number eventually reaches either this expected number or an upper bound, specified to avoid excessive running time. If the best score is sufficiently good when this happens, the respective candidate is chosen as the pose.

Given a set of keygraph correspondences Γ , a straightforward procedure for estimating the pose is applying RANSAC to the correspondences defined in Equation (16.2) or over a set of frequent keypoint correspondences obtained with a voting table [26, 27, 35]. However, both procedures discard

a valuable information: if a keygraph correspondence is correct, then all keypoint correspondences it implies are correct simultaneously. Therefore a better approach is to apply RANSAC to a family of sets as Equation (16.1). This reduces significantly the expected number of RANSAC iterations as evidenced by the results shown in Sec. 16.4.1.

Table 16.4 shows the RANSAC algorithm for estimating the pose using keygraphs.

Table 16.4. RANSAC for pose estimation using keygraphs.

```

1 let  $\Gamma$  be a set of keygraph correspondences
2 let  $k$  be number of desired keygraph correspondences
3 initialize best homography  $\mathcal{H} \leftarrow \emptyset$ 
4 initialize maximum correspondences  $mc \leftarrow 0$ 
5 initialize loop counter  $i \leftarrow 0$ 
6 while  $i < \text{loop\_limit}$  and  $mc < \text{correspondences\_threshold}$ 
7    $i \leftarrow i + 1$ 
8    $G \leftarrow \text{SAMPLE\_CORRESPONDENCES}(\Gamma, k)$ 
9    $H \leftarrow \text{COMPUTE\_HOMOGRAPHY}(G)$ 
10   $c \leftarrow \text{COUNT\_GOOD\_CORRESPONDENCES}(H, \Gamma)$ 
11  if  $c > mc$ 
12     $mc \leftarrow c$ 
13     $\mathcal{H} \leftarrow H$ 
14 return  $\mathcal{H}$ 

```

16.3.5. Combinatorial and geometric criteria

The keygraph detector is based on combinatorial and geometric criteria. Most of such criteria are applied to both model graphs and scene graphs, but some of them are applied only to the scene. First the shared criteria are described and then the section is concluded with the criteria specific to the scene.

The combinatorial criterion is the structure of the keygraphs. Figure 16.9 shows the three structures chosen for the experiments, and the respective matrices chosen for defining keytuples.

Such structures and matrices were chosen because obtaining keytuples from them is straightforward and fast: for the first structure the keytuple is unique, and for the second and third structures a keytuple is always a cyclic permutation of another keytuple. Larger circuits could have been used, but

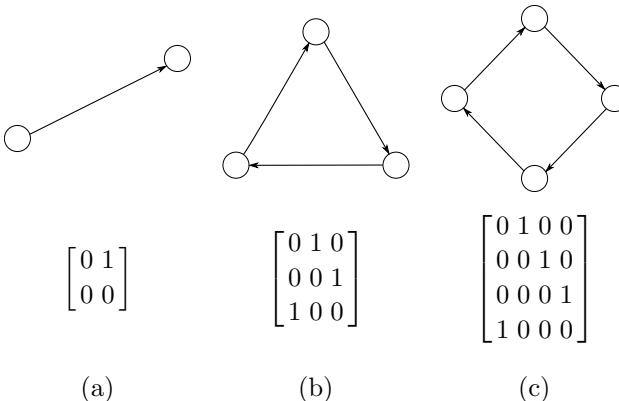


Fig. 16.9. The three structures chosen and their respective matrices. (a) Two vertices and the arc from one to the other. (b) Circuit with three vertices. (c) Circuit with four vertices.

the experimental results of Sec. 16.4.1 suggested that the possible gain in accuracy would not justify the loss in running time.

The geometric criteria, on the other hand, are three. The first criterion is minimum distance: two vertices of a keygraph cannot be excessively close according to the Chebyshev distance:

$$d_C = \max_i \{|p_i - q_i|\}$$

where p and q are two points with coordinates p_i and q_i , respectively, and $|.|$ is the absolute value.

This criterion was chosen because, as shown in Section 16.3.2, descriptors are computed from intensity profiles. Since the length of a profile is the Chebyshev distance between the respective vertices, the criterion avoids short arcs and therefore poor descriptors.

The second geometric criterion is maximum distance. This criterion was chosen to avoid long arcs and therefore the consequences of loss of locality: short intensity profiles are more robust to perspective transforms and more adequate for non-planar objects.

Finally, the third criterion is relative positioning: if the graph is one of the two circuits, its arcs must have clockwise direction. This criterion was chosen because it is assumed that mirroring is not one of the possible distortions that an instance of the object can suffer in the scene. Under this assumption, mapping a clockwise circuit to a counter-clockwise circuit does not make sense. Deciding whether a circuit is clockwise requires a straightforward and fast cross-product operation.

Note that the geometric criteria can alleviate the combinatorial explosion by reducing the number of graphs, as illustrated by Fig. 16.10.

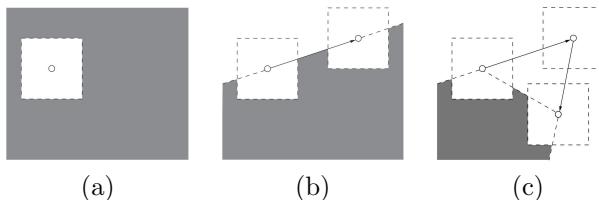


Fig. 16.10. Potential of geometric criteria for reducing the number of keygraphs. Specifically, the minimum distance and relative positioning. (a) After choosing the first vertex, the gray set of points that can be the second one is limited by the minimum distance. (b) After choosing the second vertex, the set of points that can be the third one is limited by the minimum distance and relative positioning. Note that more than a half of the Euclidean plane does not belong to the set at this stage. (c) After choosing the third vertex, the set that can be the fourth one is limited even further.

From a theoretical point of view, the aforementioned combinatorial and geometric criteria are enough to keep the number of model keygraphs tractable. Since keygraphs are isomorph, the chosen structures ensure that such number is either $O(n^2)$, $O(n^3)$, or $O(n^4)$, where n is the number of keypoints. The descriptors are stored in an indexing structure that allows correspondence selection in expected logarithmic time. Therefore, since $\log n^k = k \log n$, correspondence selection is sublinear on the number of keypoints. A superlinear number of scene keygraphs, however, is not tractable. Therefore, for those keygraphs an additional geometric criterion is chosen: they must be obtained from the edges and faces of the Delaunay triangulation [36] of the keypoints. Figure 16.11 shows an example.

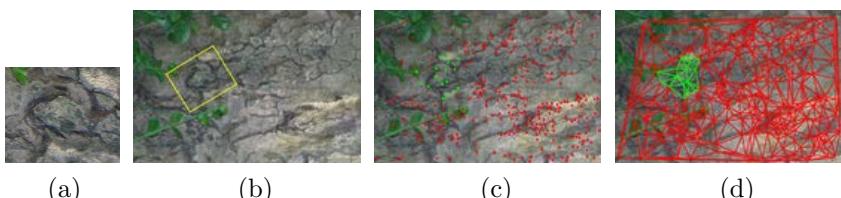


Fig. 16.11. Example of Delaunay triangulation. (a) Model. (b) Scene and pose of the instance of the object in the scene (yellow rectangle). (c) Scene keypoints: points that can be mapped to the model are emphasized (green points). (d) Delaunay triangulation of the keypoints: edges and faces that can be mapped to the model are emphasized (in green).

Since the Delaunay triangulation is a planar graph, the Euler formula can be used to prove that the number of edges and faces is $O(n)$ [36]. Furthermore, such triangulation can be obtained in $O(n \log n)$ time [37].

Note that for dense keypoint detectors, such as Hessian-Affine, the Delaunay triangulation can have an excessive number of short edges and therefore an excessive reduction on the number of keygraphs. This explains why the scene keypoints are sampled according to the minimum distance.

16.4. Applications

The performance of keygraphs for object detection was verified by applying them on two distinct applications. Firstly, the efficacy and discriminativeness of the keygraphs were compared with SIFT keypoints by performing tests on the Mikolajczyk and Schmid database [25]. In the second application, the framework was used to detect multiple objects in a real-time situation using a mobile phone.

The implementation was written in C++ and based on the OpenCV library [38]. The Delaunay triangulation was computed with the Triangle library [39] and the Fourier transform was computed with the FFTW library [40]. In RANSAC, the best candidate is refined with the Levenberg-Marquardt algorithm [41, 42].

16.4.1. Detection on general database

Two experiments were conducted. In the first experiment, image pairs manually obtained from the dataset shared by Mikolajczyk and Schmid [25] were used. This dataset is partitioned in 8 subsets: each subset has one model, five scenes, and five homographies modeling the respective poses. The five scenes represent different degrees of a specific distortion. Six subsets were chosen and, from each one, the scene representing the lowest degree of distortion and a manually cropped sub-image of the model were selected. Figure 16.12 shows the image pairs.

The objective of the experiment was to compare the accuracy of the three different structures shown in Section 16.3.5 for different images. The main interest was in discovering the best structure for low-degree distortions before considering images with high-degree distortions. The two subsets that were not chosen were *trees* (textured image with change of blur) and *ubc* (structured image with change of compression). Both were discarded for providing results excessively different: much worse for the former and

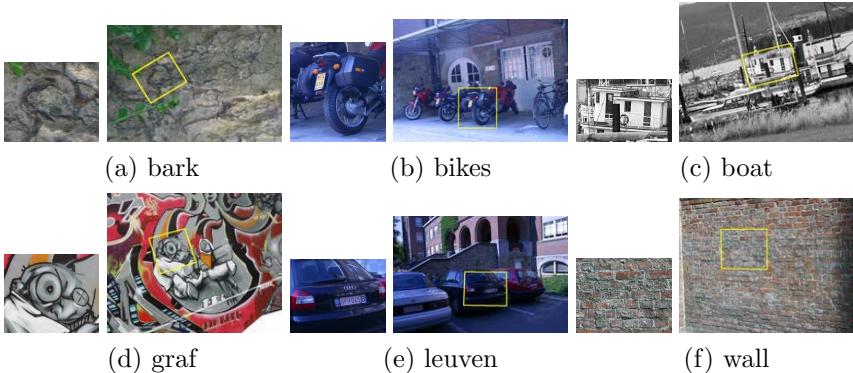


Fig. 16.12. Image pairs used in the first experiment, with visual representations of the homographies. (a) bark: textured image with change of scale and rotation. (b) bikes: structured image with change of blur. (c) boat: structured image with change of scale and rotation. (d) graf: structured image with change of viewpoint. (e) leuven: structured image with change of brightness and contrast. (f) wall: textured image with change of viewpoint.

much better for the latter.

The models were cropped to bring the dataset closer to the case of object detection in cluttered backgrounds: the cropping significantly increases the number of scene images that have no correspondence in the model, thus raising the difficulty for the textured images. The accuracy measure chosen for the evaluation was the precision-recall curve. Let $\mathcal{C} = \{(c, w)\}$ be the set of correspondences c with associated weights w obtained by the method. Let also $\mathcal{I} \subseteq \mathcal{C}$ be the subset of correct correspondences, or inliers, of \mathcal{C} . Now let $\mathcal{C}_\tau = \{(c, w) | (c, w) \in \mathcal{C} \text{ and } w > \tau\}$ be a subset of correspondences with its respective sets of inliers $\mathcal{I}_\tau \subseteq \mathcal{I}$. The recall corresponds to the ratio $|\mathcal{I}_\tau|/|\mathcal{I}|$. The precision, on the other hand, corresponds to the ratio $|\mathcal{I}_\tau|/|\mathcal{C}_\tau|$. The precision-recall curve is obtained by systematically varying the value of the threshold τ and by plotting the respective values of recall and precision for each \mathcal{C}_τ .

Since there was also interest in comparing the keygraph framework with the keypoint framework, the precision-recall curve for individual keypoint correspondences was also plotted. The keypoint descriptor chosen to select those correspondences was SIFT. In order to avoid bias for the keygraphs, it was established that the recall is always computed from keypoint correspondences: in the case of keygraphs, such correspondences are obtained from Equation (16.2).

Finally, it was also established that a keygraph correspondence was correct if all its implied keypoint correspondences were correct, and a keypoint correspondence was correct if the Euclidean distance from the groundtruth was less than 3 pixels. This value was chosen empirically. Figure 16.13 shows the results for two sequences of measurements: one with SIFT descriptors on vertices and one with Fourier descriptors on arcs. For both sequences the keypoint detector was the Hessian-Affine algorithm and the dissimilarity measure was the Euclidean distance.

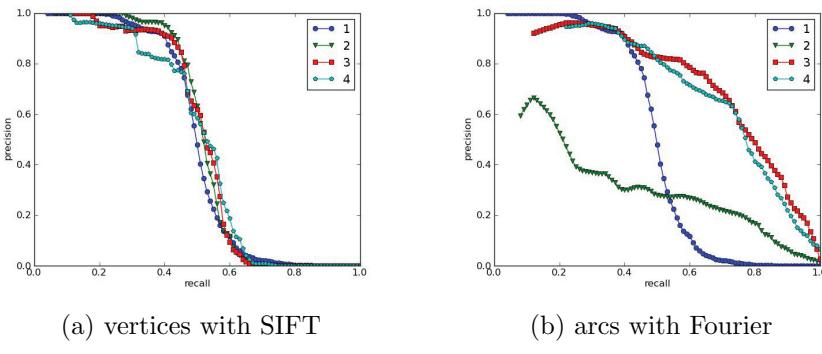


Fig. 16.13. Results for the two sequences of measurements in the first experiment: (a) for vertices with SIFT and (b) for arcs with Fourier. The name of each curve represents the number of vertices in the structure: the first one is the baseline curve for individual keypoint correspondences, hence the same in both sequences, and the other three are for the three structures shown in Section 16.3.5. Each curve is the mean of the curves across all pairs of images.

For vertices with SIFT, the similarity of all curves shows that naively using a state-of-the-art keypoint descriptor in the keygraph framework is not necessarily the best approach. For arcs with Fourier descriptors, it was observed that keygraphs consistently had worse precisions for lower recalls, but consistently had better precisions for higher recalls. This was true even for the simplest structure with two vertices, which does not have the advantages of relative positioning, although its overall precision was much worse than the others. It was also observed that the curves for the two circuits were similar. For that reason, larger circuits were not considered, as the difference in accuracy did not seem to justify the increase in running time.

Because of its balance between accuracy and efficiency, circuits with three vertices were chosen to represent keygraphs in the second experiment. In that experiment, the objective was comparing exhaustively the keygraph

framework with the keypoint framework. The previous dataset and the idea of cropping were kept, but all 5 levels of all 8 distortions were used and instead of cropping manually, 10 models for each subset were randomly cropped. Figure 16.14 shows the models and scenes for one of the subsets.



Fig. 16.14. Models and scenes for the subset graf in the second experiment.

Furthermore, a sequence of measurements was made with the Hessian-Affine detector and another with the MSER detector. In total, 800 measurements were made and their results are shown in Fig. 16.15. This second experiment confirmed that keygraphs provide superior precision for higher recalls. However, the specific recall where the curves intersect each other was not stable across subsets and detectors. Figure 16.16 shows that for two subsets the keygraphs performed consistently better with Hessian-Affine. In contrast, Fig. 16.17 shows that for other two subsets the keygraphs performed consistently worse with Hessian-Affine. With MSER the results for the same subsets were more similar to the average results, as illustrated by Fig. 16.18.

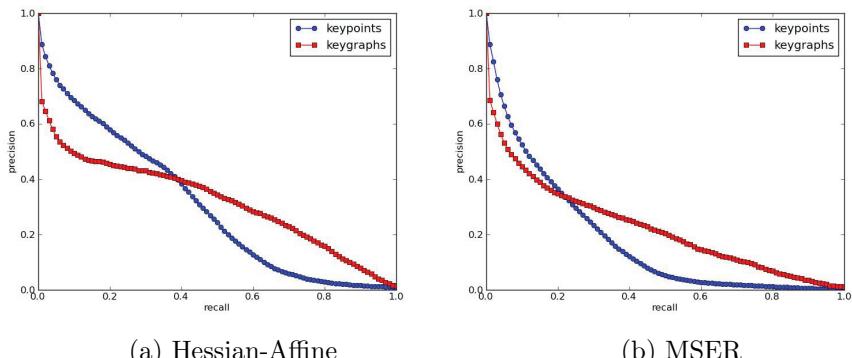


Fig. 16.15. Results for the two sequences of measurements in the second experiment: (a) for Hessian-Affine and (b) for MSER. Each curve is the mean of the 200 measurements taken for each keypoint detector.

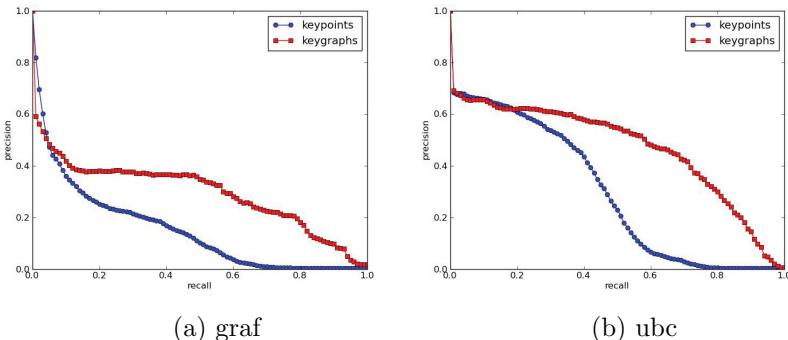


Fig. 16.16. Results for the second experiment, specifically for Hessian-Affine and two subsets: (a) for graf and (b) for ubc. Each curve is the mean of 50 measurements.

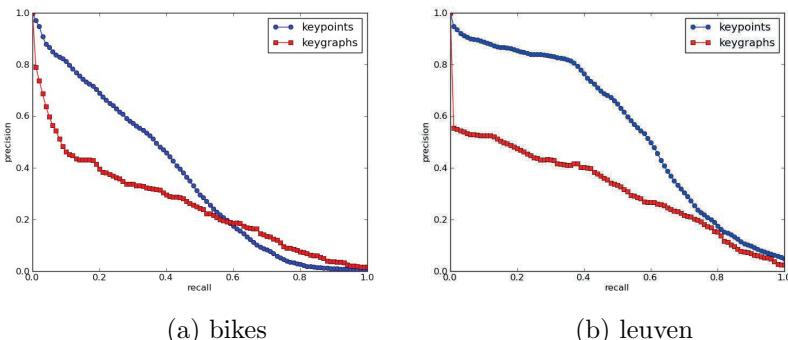


Fig. 16.17. Results for the second experiment, specifically for Hessian-Affine and two subsets: (a) for bikes and (b) for leuven. Each curve is the mean of 50 measurements.

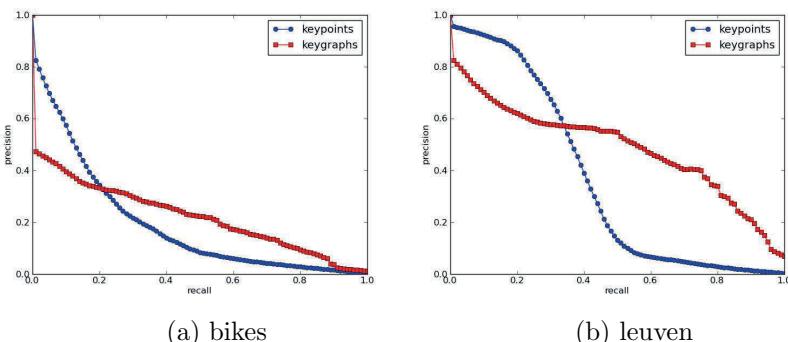


Fig. 16.18. Results for the second experiment, specifically for MSER and two subsets: (a) for bikes and (b) for leuven. Each curve is the mean of 50 measurements.

These differences are particularly interesting because they are direct consequences of the advantages of each keypoint detector. In the comparative study of Mikolajczyk *et al.* [24], Hessian-Affine performed worse precisely on the bikes and leuven subsets. This shows that the keygraph framework is directly influenced by the repeatability of the keypoint detector chosen, but when good repeatability is ensured, then keygraphs present better precision at higher recalls.

However, that does not mean that detectors with worse repeatability should be necessarily dismissed. When using RANSAC, more important than the precision p is the expected number of iterations, that can be approximated by

$$\frac{\log(1 - \mathbb{P})}{\log(1 - p^h)}$$

where \mathbb{P} is the expected probability of convergence and h is the size of the set of correspondences necessary for estimating a pose candidate. The value of h is the fundamental difference between keypoints and keygraphs: for estimating an homography, four keypoint correspondences are needed. Therefore, $h = 4$ for keypoints and $h = 2$ for circuits with three vertices. The difference in number of iterations can be seen in Fig. 16.19. In fact, even for the subsets where keygraphs had a consistently worse precision, they still had a consistently better number of iterations (Fig. 16.20).

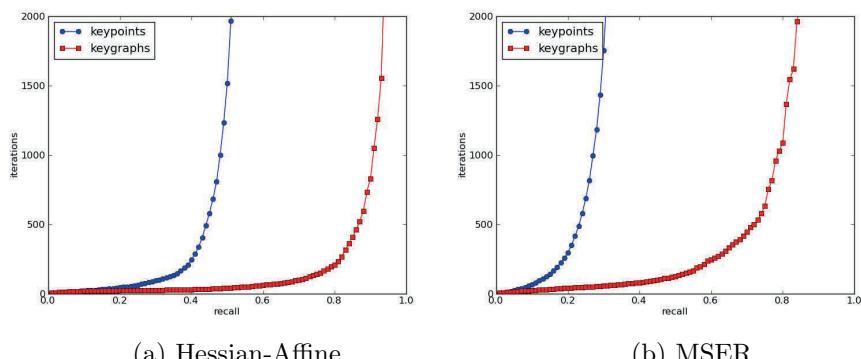


Fig. 16.19. Results for the two sequences of measurements in the second experiment: (a) for Hessian-Affine and (b) for MSER. Each curve is the mean of the 200 measurements taken for each keypoint detector.

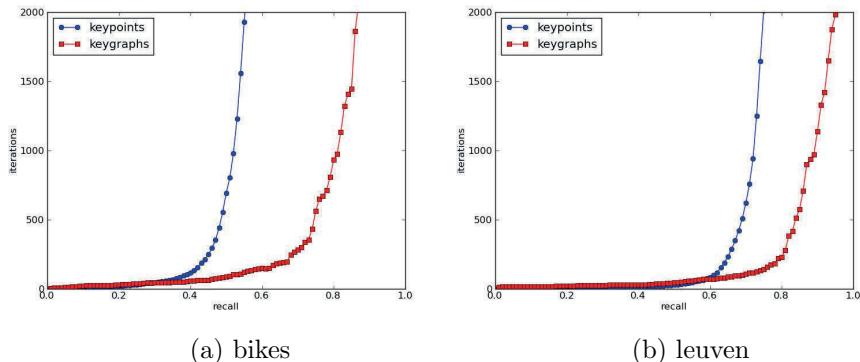


Fig. 16.20. Results for the second experiment, specifically for Hessian-Affine and two subsets: (a) for bikes and (b) for leuven. Each curve is the mean of 50 measurements.

16.4.2. *Detection on mobile phones*

To evaluate the method on devices with very restricted resources, an application for object detection in a mobile phone was developed. The phone was a Nokia N900 phone. The device was equipped with an ARM Cortex A8 600MHz processor, 256MB of RAM, and Maemo OS.

To perform the detection in real-time on the mobile phone, it was necessary to restrict some of the parameters involved:

- Model size: the models were restricted so that all of them had no more than 500 pixels on the largest side.
 - Keypoint detector: the keypoints were generated by the MSER [10] detector implemented in OpenCV. The parameters of the detector were tuned to obtain good performance on the mobile phone. As a consequence, an average of around 30 keypoints were obtained for each frame.
 - Scene size: scenes were captured in the standard VGA resolution 640 x 480.
 - Model description: keygraphs were extracted from a full keypoint set Delaunay triangulation and also from 10 additional triangulations generated using half of the points, which were chosen randomly.
 - Temporal constraint: The temporal aspect of the video was used to avoid a new search for other models every time the previous model kept more than 50% of the previous correspondences. Therefore, the system tracked a single model at a time.

The experiments were conducted in a local indoor environment using the mobile application to detect signs, posters, books, and similar objects. The detection robustness of the keygraphs as well as their scalability, i.e., how an increase in the number of models affected detection performance, were evaluated. For comparison, a robustness test was run using regular SIFT and SURF keypoints to obtain a performance reference.

16.4.2.1. Detection robustness

The keygraphs were evaluated on mobile phones by using them to detect three signs while a person walked along a corridor holding a mobile phone. To obtain more data for evaluation and more unbiased results, videos were taken by five different individuals that were not involved in the project. This test is interesting to verify the robustness of the method against adverse conditions found in real situations, such as camera limitations and motion blur.

The videos were evaluated by computing the number of frames in which a sign was detected. Simply analyzing the ratio between detected frames and the whole video is not interesting, however, because many times the signs are too far away and they are really not expected to be detected. Therefore, the frames of the videos were manually classified into *close* frames and *far* frames. A frame f was considered *close* when the ratio $r(f) = s/t$ was higher than a threshold τ , where s was the area of the sign on the picture and t was the total area of the image. The threshold τ was obtained by applying multiple detectors to the video sequences and estimating the lower bound for r at which the signs could be correctly recognized. More formally, let K_{id} be the set of indices of the frames in which a sign was detected in a video i by a detector $d \in \{\text{keygraphs}, \text{SIFT}, \text{SURF}\}$. Let also $F_{ijd} = \{f_{ijdk}\}$ be the set of frames indexed by $k \in K_{id}$ where the sign j is detected. The minimum ratio for each sign in each video can then be estimated by $\mathcal{R} = \bigcup_{i,j,d} (\mathcal{R}_{ijd})$, where $\mathcal{R}_{ijd} = \min_{k \in K_{id}} (r(f_{ijdk}))$ that, in turn, gives the threshold $\tau = \frac{1}{|\mathcal{R}|} \sum_{r \in \mathcal{R}} r$. Table 16.5 shows the values estimated for each detector and the final threshold obtained.

Table 16.5. Ratio statistics for each detector and global values. The chosen threshold τ is marked in bold.

	Min.	Mean	St. dev.
Keygraphs	0.029	0.092	0.048
SIFT	0.008	0.053	0.048
SURF	0.016	0.058	0.044
Global	0.008	0.068	0.049

The results were obtained using the default parameters of SIFT and SURF detectors and descriptors as implemented on OpenCV. They showed that, compared to SIFT or SURF, the keygraphs usually needed to be closer to an object before the object could be detected. It is important to highlight that, as described in the following section, these detectors were significantly more time-consuming than the keygraphs. Also, according to the distance function in [43] the difference between the minimum mean r values corresponds to a distance of only 60 cm in a real environment, which is not a great restriction when walking around with a mobile phone. Besides being able to detect the object from afar, it is also important to evaluate if the detector can keep a constant and reliable detection rate while the user walks around. In that sense, the three detectors were evaluated by trying to find the signs only in the *close* frames. The observed results are shown in Table 16.6.

Table 16.6. Detection performance of each detector on the videos. The detection rate is computed by considering only the *close* frames.

Total frames	5535
<i>Close</i> frames	2553
Keygraph detections	1358
Keygraph detection rate	0.531
SIFT detections	1675
SIFT detection rate	0.656
SURF detections	1127
SURF detection rate	0.441

As it can be seen by the results, the SIFT detector had the overall best stability, but the keygraphs outperformed SURF in this task. The superiority of the SIFT results over the keygraphs can be explained by the capacity of SIFT to detect the signs from afar and by the greater sensitiveness of MSER keypoints to (motion) blur [21]. In the latter case, however, it was observed (Fig. 16.21) that often the chosen MSER keypoint detector itself was unable to find points on the object. In such a situation, the keygraph descriptors cannot be extracted, indicating that the use of a more robust keypoint detector could yield better results. In fact, if only frames in which the object contained four or more keypoints were considered, the keygraphs would have a significant improvement, with a detection rate of 0.715.

Regarding the detection robustness, even though overall SIFT performs better, it was observed that it did not work well when the signs were being observed under a significant tilt. In such cases, while SURF also did not



Fig. 16.21. Example of object detection under motion blur. Even the MSER keypoint fails in such cases (none or too little vertices on the object), making a keygraph description unfeasible.

show good results, the keygraphs were able to continue to detect the object, even under more severe conditions. That is according to the results observed by other authors, such as in [44], where it is shown that both SIFT and SURF detectors show decreased performance under rotation, with SURF suffering more when the rotation angle is small. The pictures in Fig. 16.22 show a comparison between the three methods. Under these conditions, the

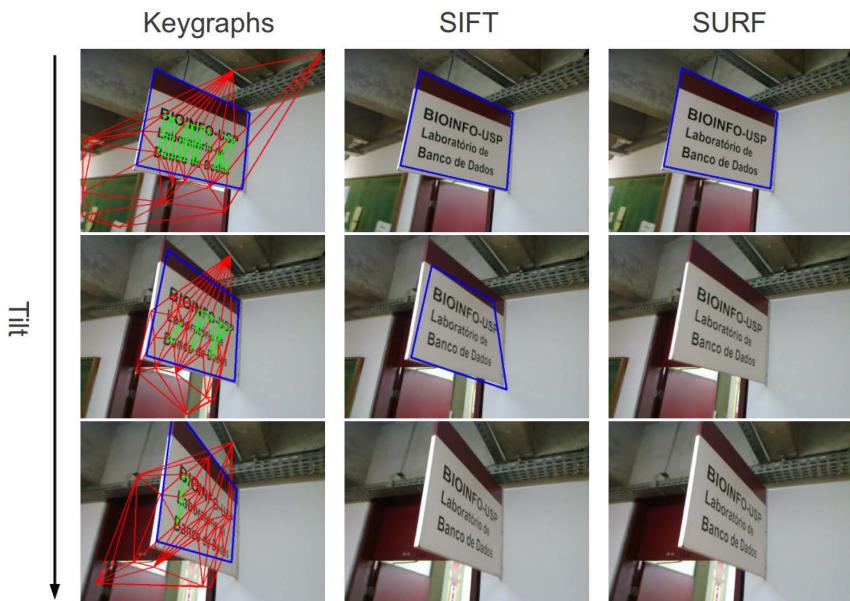


Fig. 16.22. Results showing robustness to object tilt. The left column shows results for keygraphs, the center for SIFT, and the right for SURF. The first row shows that all three work well when the object is only slightly tilted, but as the tilt increases, SURF is the first to lose track of the object, followed by SIFT. Only the keygraphs still detect the object under a significant tilt.

keygraphs had better detection performance than the other two methods. To verify that finding, the three detectors were evaluated using additional videos with tilted signs and the frames from other videos in which the object was tilted. The results, summarized in Table 16.7, confirmed that the keygraphs consistently outperformed the other detectors, showing greater robustness for object tilt. The lower performance of SIFT and SURF is consistent with the fact that both detectors assign circular neighborhoods that are not invariant to affine transformations.

Table 16.7. Evaluation of detection performance on tilted objects.

Total frames	1144
Keygraph detections	857
Keygraph detection rate	0.749
SIFT detections	619
SIFT detection rate	0.541
SURF detections	312
SURF detection rate	0.272

Figure 16.23 shows some additional results regarding the robustness of the keygraphs, in which the object is detected even under partial occlusion, and a change of scale, perspective, or lighting.

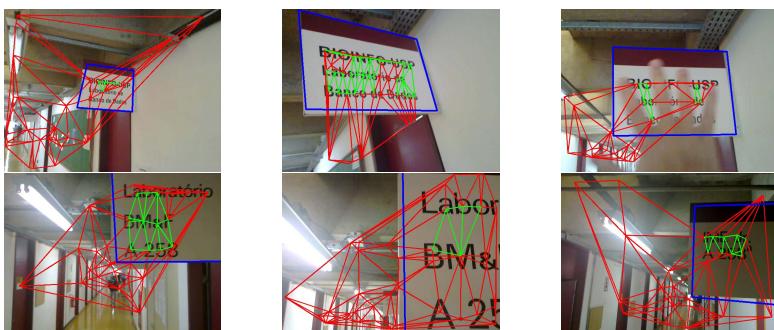


Fig. 16.23. Results showing robustness of object detection. The green edges represent detected keygraphs.

16.4.2.2. Detection time

As far as mobile devices are concerned, time performance is typically an important issue. Therefore, besides being robust, the method must also be efficient enough to be performed in the device within a reasonable time. To

evaluate the time in more detail, the whole detection process was divided into three main phases:

- Keypoint detection: time spent by the keypoint detector to obtain a set of keypoints, in this case the MSER detector.
- Descriptor computation: time necessary to obtain the descriptors. When analyzing the keygraphs performance, it is necessary to include the time to build them from the given keypoints.
- Model/Scene correspondence: is associated with the correspondence of similar keypoints or keygraphs of a scene image with one or more model candidates and finding the homography fit of the best model to estimate its pose.

Figure 16.24 shows the results obtained using keygraphs and SURF descriptors, respectively. As the main interest was measuring the time spent by each descriptor, SURF descriptors were computed over the same MSER keypoints used to build the keygraphs. As SIFT is even more time-consuming than SURF [4], it was decided to only compare the time performance with the latter.

Detection using 5 models

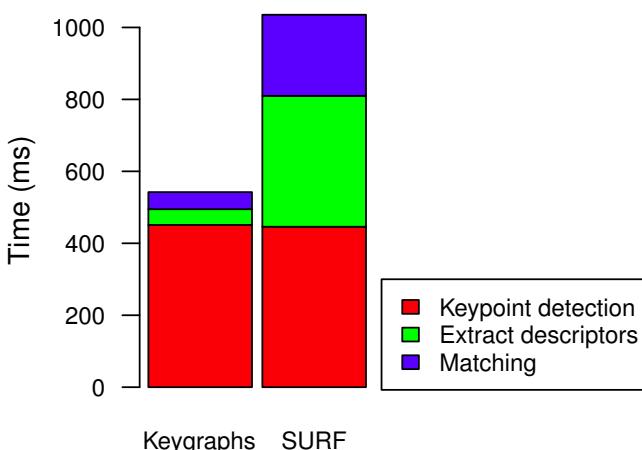


Fig. 16.24. Comparison of performance between keygraphs and SURF descriptors.

Analyzing Fig. 16.24, the first important thing to observe is that keygraph descriptors can be computed much more efficiently than SURF. Moreover, as keygraph descriptors are also much smaller (only 6 dimensions versus 64 for SURF), the performance of correspondence and pose estimation phases are also very positively affected. It should also be noticed that, when using keygraphs, most of the time is spent by the MSER keypoint detector. Thus, the overall performance could be greatly improved by simply finding a more efficient keypoint detector that could benefit from keygraph descriptors.

16.4.2.3. Scalability

It is often necessary to detect multiple objects at the same time. In that sense it is important that the method scales well, i.e., it is not hardly affected by increasing the number of models.

Figure 16.25 shows how both keygraph and SURF descriptor performances were affected as the number of models increases. As expected, the addition of more models increased the time necessary to perform detection. It is worth noting that the keygraph performance significantly improved compared to SURF as models were added. As can be seen, if only the correspondence time is considered (Fig. 16.25(b)), even when dealing with 100 models, the keygraph descriptors were able to match a scene in

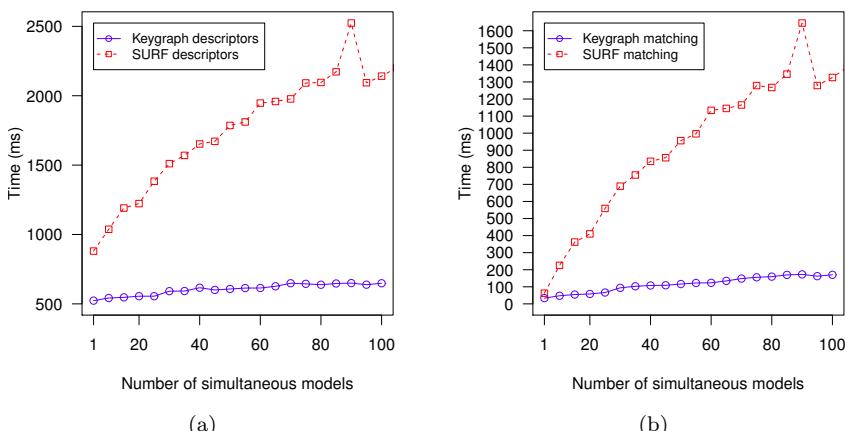


Fig. 16.25. Comparison of the scalability results between keygraph (blue) and SURF (red) descriptors. (a) time spent during the whole detection process. (b) results only for the correspondence phase, which is the only phase affected by the increase in the number of models.

around 100 ms, while SURF required more than 1 second.

The results showed that the keygraphs scale well and retain good time performance, even when dealing with many models at the same time. The time curve presented in Fig. 16.25(b) shows that the keygraph descriptor correspondence performance is comparable to state of the art binary detectors, such as [44], which reports times of around 100 ms, even using a less powerful phone.

It is important to investigate whether the addition of so many models hinders the actual object detection. To verify whether multiple objects can be detected, an experiment in a local building was performed. One hundred objects were selected and used as models for the application (Fig. 16.26 shows some of the selected objects). Afterwards, a video containing around 50 of the selected objects was recorded using the same mobile phone. This video was used as input for the detection method to check whether all of the objects could be detected. A video showing the detection results is available on: <http://youtu.be/5oWkxdVU1EU>. A few of these results can be seen in Fig. 16.27. As it can be seen from the results, most of the objects were successfully detected, with some occasional failures. The analysis of the results indicates that 43 objects were detected and 10 of them failed, yielding a considerable hit ratio of 0.81.

One of the reasons for some of the detection failures is that, as the ap-

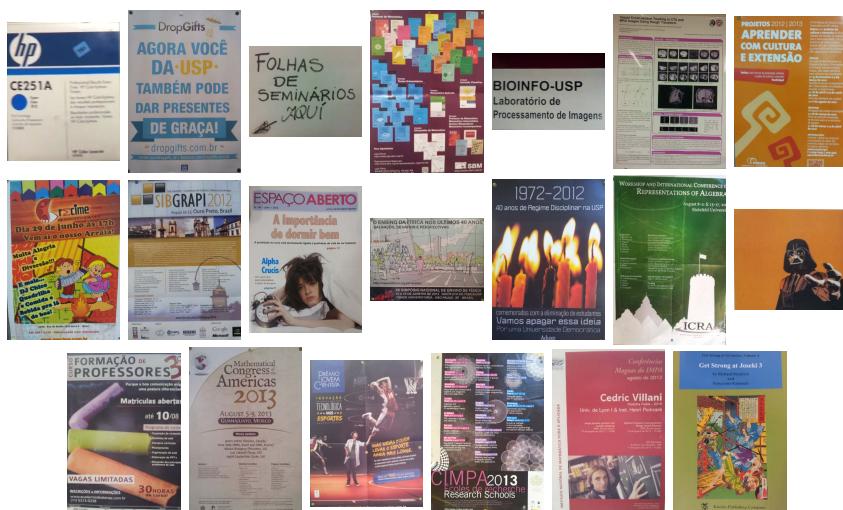


Fig. 16.26. Some of the object models of the database.

plication was built to detect a single object at a time, the occluded ones were sometimes detected instead. Another issue is the presence of motion blur, commonly seen in videos, that also affected the results. Despite the occasional failures in detection, it is interesting to point out that the application did not return any false positives, i.e., matched the wrong model. Of course, these results could be changed by decreasing the acceptance threshold for the matching, but even using a high threshold, the detection rate was acceptable.

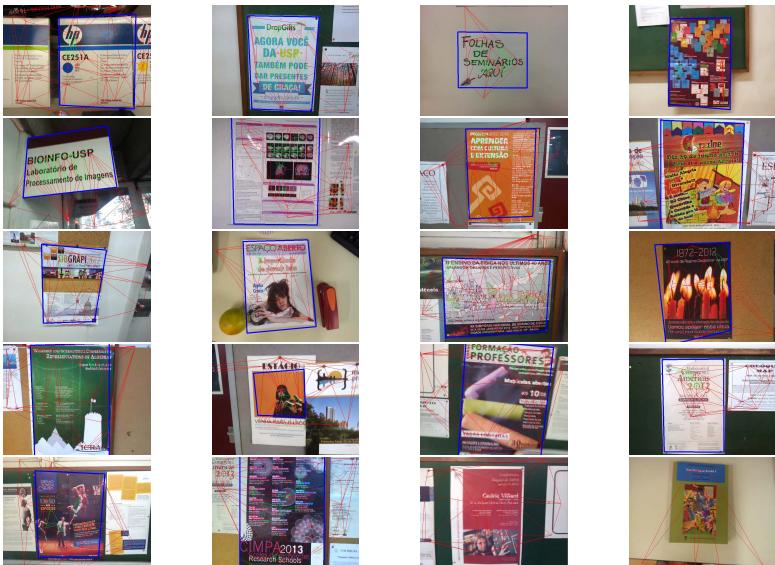


Fig. 16.27. Results from detection. Most of the objects were successfully detected (blue rectangle), with only a few occasional failures.

16.5. Conclusions

In this chapter, a framework for object detection based on a generalization of keypoint correspondences into keygraph correspondences was proposed. The main difference of this approach to similar ones in the literature is that each step deals directly with graphs: the description does not need point region descriptors, the comparison takes the entire structure into consideration, and the pose estimation uses graph correspondences directly instead of converting them with a voting table. In order to show the viability of the framework, an implementation with low theoretical complexity was ex-

plained and experiments proved both its accuracy and speed either on a standard public database and on a mobile phone application.

Acknowledgements

This research was supported by FAPESP (grants #2007/56288-1, #2011/50761-2 and #2012/09741-0), CAPES, CNPq, FINEP and PRP-USP.

References

- [1] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the 7th IEEE International Conference on Computer Vision*, vol. 2, pp. 1150–1157, IEEE Computer Society Press (1999).
- [2] D. Lowe, Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision*. **20**(2), 91–110 (2004).
- [3] V. Lepetit and P. Fua, Keypoint recognition using randomized trees, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **28**(9), 1465–1479 (2006).
- [4] H. Bay, T. Tuytelaars, and L. van Gool, SURF: Speeded Up Robust Features, *Computer Vision and Image Understanding*. **110**(3), 346–359 (2008).
- [5] J. Morel and G. Yu, ASIFT: a new framework for fully affine invariant image comparison, *SIAM Journal on Imaging Sciences*. **2**(2), 438–469 (2009).
- [6] H. Moravec. Obstacle avoidance and navigation in the real world by a seeing robot rover. Technical report, Robotics Institute, Carnegie-Mellon University (1980).
- [7] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In ed. P. J. Hayes, *Proceedings of the 7th International Joint Conference on Artificial Intelligence*, pp. 674–679, William Kaufmann (1981).
- [8] C. Tomasi and T. Kanade. Detection and tracking of point features. Technical report, School of Computer Science, Carnegie-Mellon University (1991).
- [9] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 593–600, IEEE Computer Society Press (1994).
- [10] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In eds. D. Marshall and P. L. Rosin, *Proceedings of the 13th British Machine Vision Conference*, pp. 384–396, British Machine Vision Association (2002).
- [11] E. Tola, V. Lepetit, and P. Fua, DAISY: an efficient dense descriptor applied to wide baseline stereo, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **32**(5), 815–830 (2010).
- [12] C. Schmid and R. Mohr, Local grayvalue invariants for image retrieval, *IEEE*

- Transactions on Pattern Analysis and Machine Intelligence.* **19**(5), 530–535 (1997).
- [13] M. Agrawal, K. Konolige, and M. R. Blas. CenSurE: Center Surround Extremas for realtime feature detection and matching. In eds. D. Forsyth, P. Torr, and A. Zisserman, *Proceedings of the 10th European Conference on Computer Vision: Part IV*, vol. 5305, *Lecture Notes in Computer Science*, pp. 102–115, Springer (2008).
 - [14] L. F. Costa and R. M. Cesar-Jr., *Shape classification and analysis: theory and practice*, 2nd edition. CRC Press (2009).
 - [15] R. Diestel, *Graph Theory*, 4th edition. Springer Verlag (2010).
 - [16] R. Sedgewick, *Algorithms in C, Part 5: Graph Algorithms*, 3rd edition. Addison-Wesley Professional (2001).
 - [17] P. Wei, Y. Zhao, N. Zheng, and S.-C. Zhu. Modeling 4D human-object interactions for event and object recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3272–3279 (2013).
 - [18] B. Yao and L. Fei-Fei. Action recognition with exemplar based 2.5D graph matching. In *Proceedings of the European Conference on Computer Vision*, pp. 173–186. Springer (2012).
 - [19] Y. Avrithis and K. Rapantzikos. The medial feature detector: Stable regions from image boundaries. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1724–1731 (2011).
 - [20] A. Oliva and A. Torralba, Modeling the shape of the scene: A holistic representation of the spatial envelope, *International Journal of Computer Vision*. **42**(3), 145–175 (2001).
 - [21] T. Tuytelaars and K. Mikolajczyk, Local invariant feature detectors: a survey, *Foundations and Trends in Computer Graphics and Vision*. **3**(3), 177–280 (2008).
 - [22] R. M. Cesar Jr., E. Bengoetxea, I. Bloch, and P. Larranaga, Inexact graph matching for model-based recognition: evaluation and comparison of optimization algorithms, *Pattern Recognition*. **38**(11), 2099–2113 (2005).
 - [23] K. Mikolajczyk and C. Schmid, Scale & affine invariant interest point detectors, *International Journal of Computer Vision*. **60**(1), 63–86 (2004).
 - [24] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. van Gool, A comparison of affine region detectors, *International Journal of Computer Vision*. **65**(1–2), 43–72 (2005).
 - [25] K. Mikolajczyk and C. Schmid, A performance evaluation of local descriptors, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **27**(10), 1615–1630 (2005).
 - [26] D. Tell and S. Carlsson. Wide baseline point matching using affine invariants computed from intensity profiles. In eds. A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, *Proceedings of the 6th European Conference on Computer Vision: Part I*, vol. 1842, *Lecture Notes in Computer Science*, pp. 814–828, Springer (2000).
 - [27] D. Tell and S. Carlsson. Combining appearance and topology for wide baseline matching. In ed. D. Vernon, *Proceedings of the 7th European Conference on Computer Vision: Part I*, vol. 2350, *Lecture Notes in Computer Science*,

- pp. 68–81, Springer (2002).
- [28] J. E. Bresenham, Algorithm for computer control of a digital plotter, *IBM Systems Journal*. **4**(1), 25–30 (1965).
 - [29] M. Ebrahimi and W. W. Mayol-Cuevas. SUSurE: Speeded Up Surround Extrema feature detector and descriptor for realtime applications. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 9–14, IEEE Computer Society Press (2009).
 - [30] C. Silpa-Anan and R. Hartley. Optimised KD-trees for fast image descriptor matching. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE Computer Society Press (2008).
 - [31] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In eds. A. Ranchordas and H. Araújo, *Proceedings of the 4th International Conference on Computer Vision Theory and Applications*, vol. 1, pp. 331–340, INSTICC Press (2009).
 - [32] Y. Ke and R. Sukthankar. PCA-SIFT: a more distinctive representation for local image descriptors. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 506–513, IEEE Computer Society Press (2004).
 - [33] C. G. Harris and M. J. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pp. 147–152, University of Sheffield Printing Unit (1988).
 - [34] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In eds. A. Leonardis, H. Bischof, and A. Pinz, *Proceedings of the 9th European Conference on Computer Vision: Part I*, vol. 3951, *Lecture Notes in Computer Science*, pp. 430–443, Springer (2006).
 - [35] Y. Kanazawa and K. Uemura. Wide baseline matching using triplet vector descriptor. In *Proceedings of the 17th British Machine Vision Conference*, vol. I, pp. 267–276, British Machine Vision Association (2006).
 - [36] M. de Berg, O. Cheong, M. van Krefeld, and M. Overmars, *Computational Geometry: Algorithms and Applications, Third Edition*. Springer (2008).
 - [37] S. Fortune. A sweepline algorithm for Voronoi diagrams. In *Proceedings of the 2nd Annual Symposium on Computational Geometry*, pp. 313–322, ACM Press (1986).
 - [38] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media (2008).
 - [39] J. R. Shewchuk. Triangle: engineering a 2D quality mesh generator and Delaunay triangulator. In eds. M. C. Lin and D. Manocha, *Applied Computational Geometry: Towards Geometric Engineering*, vol. 1148, *Lecture Notes in Computer Science*, pp. 203–222, Springer (1996).
 - [40] M. Frigo and S. G. Johnson, The design and implementation of FFTW3, *Proceedings of the IEEE*. **93**(2), 216–231 (2005).
 - [41] K. Levenberg, A method for the solution of certain non-linear problems in least squares, *Quarterly of Applied Mathematics*. **2**(2), 164–168 (1944).
 - [42] D. Marquardt, An algorithm for least-squares estimation of nonlinear parameters, *SIAM Journal on Applied Mathematics*. **11**(2), 431–441 (1963).

- [43] H. Morimitsu, M. Hashimoto, R. B. Pimentel, R. M. Cesar-Jr., and R. Hirata-Jr. Keygraphs for sign detection in indoor environments by mobile phones. In eds. X. Jiang, M. Ferrer, and A. Torsello, *Graph-Based Representations in Pattern Recognition*, vol. 6658, *Lecture Notes in Computer Science*, pp. 315–324, Springer Berlin / Heidelberg (2011).
- [44] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to SIFT or SURF. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2564–2571 (2011).

Chapter 17

Mining Multimodal Data

Santanu Chaudhury[^], Lipika Dey*, Ishan Verma* and Ehtesham Hassan*

[^]*Department of Electrical Engineering*

Indian Institute of Technology, Delhi

schaudhury@gmail.com

**Innovation Labs, Tata Consultancy Services, Delhi, India*

{lipika.dey,ishan.verma,ehtesham.hassan}@tcs.com

Multimodal data mining refers to analyzing more than one form of data for discovering hidden patterns. multimodal data analytics offer immense opportunities to integrate text, visual, audio, sensor data and structured information to derive and validate insights none of which may be possible to obtain from any single source. In this chapter, we have examined different aspects of dealing with big multimodal data. We have presented an example of cross-modal information integration between text and image. We have described an application of multi-structured data mining system for business analytics.

17.1. Introduction

Multimodal data mining refers to analytical paradigms that aim at simultaneously exploiting more than one of a multitude of ways that may be used to represent underlying information, namely image, video, speech, text, sensor readings or numeric variables for different measurable parameters. Multimodal data mining has been a popular research area since 2006, where the focus has been primarily to retrieve information from multimedia databases, which contain images tagged with text. Information retrieval aims at retrieving data that is relevant to a user query. The scope of multimodal data analytics however need not be restricted to information retrieval only and therefore need not start with a query. In this article we take a look at the wider scope that exists for multimedia data analytics, much of which is

admittedly unexplored.

With the proliferation of freely available data from multiple sources, multimodal data analytics offers huge possibilities of gathering insights through integration of a wide-variety of data coming from heterogeneous sources. An unrestricted multimodal data mining framework need not make any assumptions about how, where and when the data were generated. Intelligent analytical methods should be able to extract, aggregate and decide about the right levels of granularity at which information from heterogeneous sources can be fused in order to generate the desired insights. A framework such as this offers immense opportunities to integrate text, visual and structured information to derive and validate insights none of which may be possible to obtain from any single source. Increasing volume and variety of data originating from autonomous and distributed sources make the problem more challenging.

Let us highlight this through a few examples. Multimodal News analytics refers to joint analysis of the text contained in the News articles and the videos or images that may or may not be directly associated with the article but contains information about the same event. Event-based News analytics try to identify a few key events for any major event such that the key events may be strewn together to build a more structured representation of the major event. The major advantage of using multiple data sources is to be able to integrate multiple perspectives about the same event. In a slightly different scenario, it is possible to generate insights about business data using information extracted from associated text data sources. For example, enterprise business data like sales data or market share reports can be linked to relevant text data that may be discussing the organization or product. The linkages can help extract rich causal insights, which are otherwise impossible to derive.

Data mining in medical domain also requires exploration of multiple types of data. A patient data consists of the patient's physical parameters like age, height, weight and demographic information such as gender, age, family disease history, and so on. With this we need to consider data from a variety of imaging modalities like X-ray, USG, CT-Scan, MRI, PET etc. We cannot also ignore data from a DNA or genomic-related tests. Microarray expression images and sequences are used to represent the genetic code information. To create model of, for example, prognosis of a disease we need to aggregate these multimodal medical data, possibly obtained at fixed temporal intervals. Geography can introduce additional dimension to these parameters. Complex relationship between different types of data

and evolution of data has to be considered for discovering useful patterns from Big Medical Data collections.

This chapter is focused on multimodal data mining. In the next section, we examine different aspects of dealing with big multimodal data. In Section 17.3, we have presented a scheme for cross-modal information integration between text and image. In Section 17.4, we have described a multi-structured data mining system for business analytics.

17.2. Dealing with Multimodal Big Data

Some distinct challenges of Big Data are linked to heterogeneous data types, complex intrinsic semantic associations in these data, and complex relationship networks among data. The main motivation for discovering knowledge from massive multimodal data is improving the efficiency of single-source mining methods. Since, massive data are typically collected from different data sources providing different types of data, the knowledge discovery with the big data must be performed using a multisource mining mechanism. Big, heterogeneous and real-time characteristics of multisource data provide essential differences between single-source knowledge discovery and multisource data mining. It is an open problem to efficiently describe semantic features and to build semantic association models to bridge the semantic gap of various heterogeneous multimodal data sources. In contrast to static data, data streams are widely used in financial analysis, online trading, and medical testing, and so on. Static knowledge discovery methods cannot adapt to the characteristics of dynamic data, such as continuity, variability, rapidity. A nice overview of data mining with big data was presented in [1]. In this section we shall briefly review issues of multimodal data analytics.

Text combined with images provides clarity in expressing inherent semantics. Construction of models linking images and texts generated from independent sources can be used effectively for automated image annotation. Web based image collections and available annotations provide data for learning a model for predicting words with high posterior probability given an image. Examples of automated annotation appear in Barnard and Forsyth [2, 3]. A tool that could automatically suggest images to illustrate blocks of text from an image collection can be of great help for authors. Auto-illustration is possible if one can obtain images with high probability given text [2]. Learning the association between image regions and semantic correlates (words) is an interesting example of multimodal data mining,

particularly because it is typically hard to apply data mining techniques to large collections of images. In Section 17.3 we have discussed one such application.

An example of multimodal big data is Spatial Data. Spatial data describes spatial objects' specific geographic information, its spatial distribution and its relation with other spatial objects. The data can be height of a location, road length, polygonal area and building volume. It can be the string of geographical name and annotation. It also includes graphics, images, spatial relationships and topologies. Data is collected by using macro and micro sensors or devices such as radar, infrared, optical, satellite multispectral scanner, imaging spectrometer, telescopes, electron microscopy imaging and CT imaging. Spatio-temporal evolution and volume are the key attributes of spatial data collections [4, 5]. The research and development of big data applications targeted for national security, climate variation, disaster prevention and reduction are all associated with multimodal spatial data. Basic challenges in these applications are to identify data types most appropriate for a problem and designing fusion schemes for discovering hidden data patterns. For example, in the Google Earth based Rainfall disaster monitoring system, users use the Google Earth 3D terrain image, superimposing the satellite cloud images, rainfall map, single station rainfall data, soil data, and live pictures to get the three-dimensional visualization disaster effect, to analyze submergence and do decision-making [4]. In [6] a system is proposed for disaster event situation awareness. The key technology used for this research is the Context Discovery Application (CDA) [7], which is a geo-visual analytic environment. The system is designed to integrate implicit geographic information. It can deal with geographic information embedded in the text in the form of RSS feed. As Earth Observation data is growing exponentially, we need new ways to more efficiently exploit big data archives. In [8] a tool has been proposed which uses the unsupervised clustering results of different number of features, extracted from different raster image types and from existing GIS vector maps. As an outcome of clustering, each image is represented in the form of a Bag of words. In the online part, the user can introduce positive and negative examples about the specific semantic he/she is interested in and can search by similarity matrix or probability of the label in the archive. Spatial knowledge discovery techniques use spatial data mining methods to extract previously unknown, potentially useful knowledge. By using new knowledge, data can be analyzed in real time to make intelligent judgments and well-informed decisions [4].

While considering mining of multimodal data, one interesting aspect to deal with is the temporal information. This becomes particularly relevant for mining with large multimedia data collections. Multimedia data are unstructured and data is processed to arrive at content information, like event label. Such processing often leads to non-unique results with several possible interpretations. Further, the data are often the result of outputs from various kinds of sensor modalities with each modality requiring different preprocessing, synchronization and transformation procedures. Inherent uncertainty associated with labels associated with different modalities necessitates usage of probabilistic framework. As an example [9], consider a scenario in which three event detectors T₁, T₂, T₃ are labeling three different events say A, B and C at different times by extracting features from three different modalities. Each observation can be represented as (Event label, Event Probability, Start time, End time). The basic problems of data mining with this representation are [9]:

- (a) How to use associated confidence or probability of the semantic tags?
- (b) How to exploit correlation among the various media streams?
- (c) How to synchronize outcome of different detectors based on media stream they utilize? For example, the face identification may use one frame on video data in contrast to the speaker recognizer system identifying person using audio data.

More accurate and useful knowledge can be discovered with multimedia temporal sequences by using probabilistic weighting as proposed in [9] compared to the deterministic approaches proposed in [10, 11].

17.3. Multimodal Concept Linkage using Conditioned Topic Modeling

In practice, various multimedia applications deal with data having information in multiple modes. In [15], a MKL based methods for text document retrieval having multimodal information has been presented. The performance of concept-based retrieval system significantly depends on the understanding of the semantics of content. The multimodal information is mostly complementary in nature. In case of availability of such information, semantic understanding of the content can be improved by exploiting the correspondence information. In this context, concept level modality fusion provides logical approach for multimodal content retrieval. The following

discussion presents framework for learning the concept level semantic correlation between document modalities by applying combination of generative and discriminative modeling. Availability of concept lexicon is not always guaranteed. Here, the topic models for text modeling provide an efficient solution [13, 14, 17]. These models explore the underlying semantic structure of a document collection and discover the semantic grouping of existing terms. In recent works, topic models have been successfully extended for image analysis. In this work, we begin with cross-modal content retrieval framework proposed in [18] and propose conditioned topic learning in LDA based indexing to learn the latent topic assignment aligned with their contextual grouping.

17.3.1. *Conditioned Topic Learning for Multimodal Retrieval*

The framework presents generative-discriminative modeling approach for indexing the multimodal documents. The model learns the multimodal concepts from the document collection by defining conditioned topic modeling method. The proposed concept applies LDA based generative model for semantic indexing of documents.

17.3.1.1. *The Topic Learning Model*

The proposed model generalizes the LDA modeling over multimodal document collection. The model here learns the relationship between latent topics generated from different modalities. Consider D as the multimodal document collection. The indexing model for the document sets assumes that each document contains a modality index y_k , i.e. the set of documents is represented as $D = \{(y_1, \mathbf{w}_1), (y_2, \mathbf{w}_2), \dots, (y_{|D|}, \mathbf{w}_{|D|})\}$. The set $\{1, \dots, V_M\}$ contains the vocabulary size of different modalities and each word w_{kn} for $(1 < n < N_k)$ assumes a discrete value from this set. Evidently, the definition of set D relaxes the condition of one-to-one correspondence between documents from different modalities. The information of context level relationship between the documents is expressed by a similarity graph $G = (D, E)$ which is computed at the topic level. The multimodal document random field model presented in [18] defines the similarity graph by considering the pair-wise potential. The present method expresses the contextual relationship between documents by applying unary and pair-wise potentials. The objective is the incorporation of broad level contextual information for the latent topic learning over multimodal document collec-

tion. The potential function for each node is defined as

$$E(\theta_i) = \exp(-\lambda_1 f_1(\theta_i) + (-\lambda_2 f_2(\theta_i, \theta_j))) \quad (17.1)$$

In this sense, the graph represents a conditional random field over the document collection. Here unary potentials are posterior probability estimate by a learned classifier using the individual topic distribution corresponding to documents from different modalities. The pair-wise potential estimates are defined as the symmetric KL divergence between topic distributions.

The generative procedure of the proposed topic modeling follows the conventional LDA generative procedure, which first samples θ_k for k^{th} document, and subsequently samples words for k^{th} document with respect to θ_k . However, the topic distribution in present scenario also considers the contextual similarity information of documents. The similarity graph G expresses the document relationship as well as contextual category information of different documents. The generative process is summarized as follows:

- Sample the word distribution for m^{th} modality as $\varphi_m \sim Dir(\varphi|\beta_m)$, that is, a Dirichlet distribution with parameter β_m .
- Sample the document-topic distribution $\theta_{1,\dots,|D|}$ as

$$p(\theta_{1,\dots,|D|} | \alpha, \mathcal{G})$$

$$= \frac{1}{Z} \exp \left\{ -\lambda_1 \sum_{k=1,\dots,|D|} f_1(\theta_k) - \lambda_2 \sum_{k,j \in \mathcal{E}} f_2(\theta_k, \theta_j) \right\} \times \\ \prod_{k=1,\dots,|D|} Dir(\theta_k | \alpha). \quad (17.2)$$

- Sample topic z_{kn} for word w_{kn} from $Mult(z|\theta)$, that is, a multinomial distribution with parameter vector θ .
- Sample word w_{kn} from $Mult(w|\varphi_{mz_{kn}})$.

The joint probability for document collection with incorporation of similar-

ity as well as contextual category information is expressed as

$$\begin{aligned}
 & p(\mathcal{D}, \boldsymbol{\theta}_{1,\dots,|D|}, z_{1,\dots,|D|}, \boldsymbol{\varphi} | \alpha, \beta_{1,\dots,|M|}, \mathcal{G}) \\
 &= \frac{1}{Z} \prod_{m=1}^M \prod_{k=1}^K \left[Dir(\varphi_{mk}, \beta_k | \alpha) \times \right. \\
 &\quad \left. \left\{ \exp \left(-\lambda_1 \sum_{k=1,\dots,|D|} f_1(\boldsymbol{\theta}_k) - \lambda_2 \sum_{k,j \in \mathcal{E}} f_2(\boldsymbol{\theta}_k, \boldsymbol{\theta}_j) \right) \right\} \right] \times \\
 &\quad \prod_{k=1,\dots,|D|} Dir(\boldsymbol{\theta}_k | \alpha) \left(\prod_{n=1}^{N_k} Mult(z_{kn} | \theta_k) Mult(w_{kn} | \phi_{yk} z_{kn}) \right) \quad (17.3)
 \end{aligned}$$

17.3.1.2. Inferencing and Parameter Estimation

The form of the Equation (17.3) shows that exact inferencing is not possible because of the intractable form of the likelihood equation due to coupling between $\boldsymbol{\theta}$ across the document collection. Following the approach presented in [12], we adopt empirical conditioned topic learning model where context level similarity enforced by conditional random field enforced by an empirical topic distribution $\hat{\boldsymbol{\theta}}_d$ for document and compute the unary and pair-wise potentials using these distributions. The empirical topic distribution $\hat{\boldsymbol{\theta}}_d$ is computed as

$$\hat{\theta}_{kl} = \frac{n_{kl} + \alpha}{\sum_{l=1}^T n_{kl} + T_\alpha}$$

Here T is the number of topics. The parameter T_α introduces smoothness in the distribution. We apply Gibbs sampling for parameter estimation in the proposed topic learning model [16]. Gibbs sampling is a special case of Markov Chain Monte Carlo (MCMC) simulation which provides simple algorithms for approximate inferencing in high-dimensional models such as LDA. The process applies latent-variable method of MCMC simulation where the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ are integrated out as they are interpreted as the statistics of associations between the observed w_{kn} and the corresponding z_{kn} , the state variable of the Markov chain. In this context, the strategy of integrating out some of the parameters for inferencing is generally referred to as collapsed Gibbs sampling. In the collapsed Gibbs sampling, $\boldsymbol{\theta}$ and $\boldsymbol{\varphi}$ are integrated out and sampling is performed for z . For k^{th} document from m^{th} modality, the probability of l^{th} topic assignment

to word is computed as:

$$\begin{aligned}
 p(z = l | \mathcal{D}, \mathbf{x}_w, \alpha, \beta) \\
 \propto & \frac{n_{kl} + \alpha}{\sum_{l=1}^T n_{kl} + T_\alpha} \cdot \frac{n_{lw} + \beta_m}{\sum_{w=1}^{V_M} n_{lw} + V_M \beta_m} \times \\
 & \left\{ \prod_{k', (k, k') \in \mathcal{E}} \exp \left\{ -\lambda_1 \{ f_1(\boldsymbol{\theta}_{k, -z}) - f_1(\boldsymbol{\theta}_{k, z=l}) \} - \right. \right. \\
 & \left. \left. \lambda_2 \{ f_2(\boldsymbol{\theta}_{k, -z}, \boldsymbol{\theta}_{k'}) - f_2(\boldsymbol{\theta}_{k, z=l}, \boldsymbol{\theta}_{k'}) \} \right\} \right\} \tag{17.4}
 \end{aligned}$$

$\theta_{k, z=l}$ is the empirical topic distribution for k^{th} document with w assigned the topic.

l , and $\theta_{k, -z}$ is the topic distribution for k^{th} document without considering word w . For each iteration of Gibbs sampling based parameter estimation, given S subject categories in the document collection, we compute S probabilities: $P(1|\hat{\boldsymbol{\theta}}_1), \dots, P(S|\hat{\boldsymbol{\theta}}_d)$. Here $P(l|\hat{\boldsymbol{\theta}}_i)$ denotes the probability of $\hat{\boldsymbol{\theta}}$ belonging to subject l . The values here are the notion of the top-down uncertainty of document content [19]. The objective here is to align the final topic assignment to most confident subject category. The unary potential represented by f_1 is estimated as $\sum_i^S \log P(i|\hat{\boldsymbol{\theta}}_d)$.

17.3.1.3. Experimental Results and Discussion

The proposed scheme is experimentally validated on the dataset discussed in [18]. The dataset contains 2600 images of variety of subjects and corresponding textual description describing its information content. Sample images are shown in Figure 17.1. A subset of 676 examples images is randomly selected and the subset is annotated in eight categories based on the subjective analysis of the image content and corresponding description. The categories basically represent the context groups defining a major subject. The details of the categories are as follows.

The text available with image present loose description of the information contained. For the text cleaning purpose, words having less than 4 characters, and words having less than 5 occurrences in the dataset are filtered out at pre-processing stage. The unary potential for the empirical topic distribution estimation is computed by Bayesian probability estimate as discussed in Section 17.1.1. The probability estimate is computed by Relevance vector classifier learned on the image annotated with the above

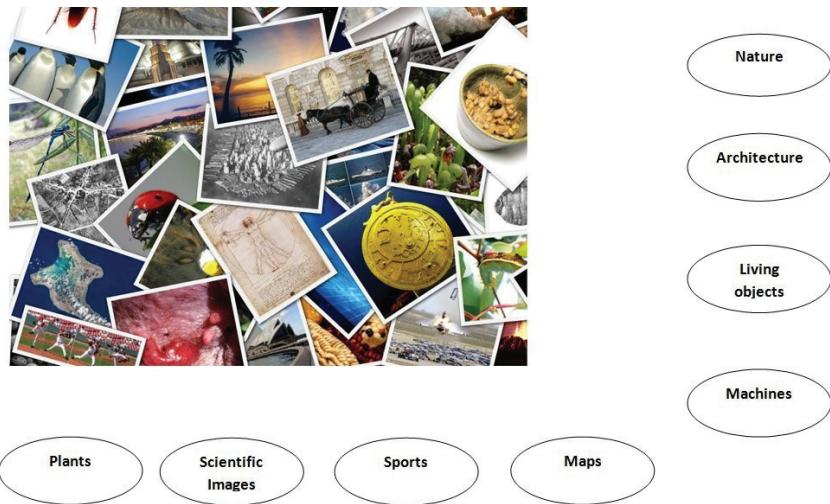


Fig. 17.1. Sample images and corresponding subject based categories.

Table 17.1. Description of different subject categories.

Broad level category	Covered subjects
Nature	Landscape, sky, oceans, mountains
Architecture	Buildings, bridges
Living objects	Humans, birds, animals, insects
Machines	cars, planes, bikes
Maps	hand-drawn and satellite images, portraits, paintings
Sports	All the sports related images
Scientific Images	Constellations, scientific equipments and description charts
Plants	Grains, vegetables and flowers

defined subject categories. The image feature representation based on bag-of-words model is computed by the SIFT descriptors. The bag-of-words based representation is computed with code-book having 1000 visual words. The methodology discussed in [18] is applied for evaluation. The performance is evaluated by retrieving the relevant images corresponding for given text queries. For text query $\mathbf{w} = \{w_1; w_2; \dots; w_n\}$, the dataset images are ranked using $p(\mathbf{w}|\boldsymbol{\theta}_i) = \prod_{j=1}^n p(w_j|\boldsymbol{\theta}_i)$.

The $\boldsymbol{\theta}_i$ represents topic distribution for the i^{th} image of the collection. The marginal probability estimate $p(w_j|\boldsymbol{\theta}_i)$ for all the text terms are computed while training. Using a synthetic query set, the presented approach

achieved 50.76% accuracy in the first 20% of ranked list of results. For the same query set, the MDRF method discussed in [18] retrieved 48.04% accuracy. Figure 17.2 shows retrieved images for two sample text queries using our, and approach presented in [18]. The results establish that conditioned topic modeling efficiently exploits the dependency between multimodal features and labels resulting more accurate semantic grouping.

<p>Abbey of Senanque, located in France, Provence, Vaucluse, Gordes village. An abbey is a Christian monastery or convent, under the government of an Abbot or an Abbess, who serve as the spiritual father or mother of the community.</p>	
<p>The Loch Ard Gorge, found in Port Campbell National Park, Victoria, Australia, is named after the clipper ship Loch Ard, which ran aground on nearby Muttonbird Island on 1 June 1878 approaching the end of a three-month journey from England to Melbourne. Shown here is a panorama of 4 segments taken from the cliffs looking down towards Loch Ard Gorge.</p>	

Fig. 17.2. Sample retrieval results corresponding to the textual description in the left: The top row shows retrieved image from the proposed method and bottom row shows the images retrieved by the method presented in [18].

17.4. Multi-structured Data Analytics for Business Intelligence

Business Intelligence analyzes business performance data of self and competing organizations to assess risks and opportunities for business based on internal and external information. Business data analyzed in isolation can only show what has happened and not why it has happened. However, performance data when linked with unstructured text data containing News or reports can yield causative insights. The causal associations may be ex-

plicitly stated in the text or implicitly learnt through association mining or other supervised and unsupervised machine learning techniques. Causative insights can produce far better predictions than forecast based only on past performance data. Data-based predictions cannot take into account environmental factors or deviations. Knowledge-based predictions are far more powerful since they take are capable of accounting for many more internal and external factors. Learning such a model is however extremely difficult and not yet a solved problem.

Market News captures important information about customers, competitors, suppliers or any other events of global political, financial or social relevance. Capability of an organization to interpret such events in conjunction with its own business results and strategies are keys to its survival in the competitive world. However, since the potential set of events drawn from the innumerable News collections is extremely large, intelligent mechanisms are needed to learn to detect the events of business interest. The events of interest can vary from company to company as well as from time to time. Multimodal big data analytical techniques promise to play an important role in linking, analyzing and learning to detect business-significant events for organizations.

Figure 17.3 presents a search-and-analytics based framework for Multi-structured big data analytics to derive predictive Business Intelligence using text information from News articles and structured business data. The framework supports data collection, analysis and linking of multi-structured data through intelligent processing of structured and unstructured data. It is an evolutionary framework that is intended to learn business-critical causations through human interactions and feedback. The functionalities of the three main components of the proposed self-learning, multimodal analytical framework are as follows:

- (i) Data collection, processing and indexing – The data collection and processing components of the framework support crawling, pre-processing, text-processing and indexing of text documents sourced in from multiple sources. Business data in this framework is assumed to be time-stamped information components, reporting performance of a measured component at a particular time. The stored data is indexed by the organization name, place, time, product or other names that are available as meta-data at the time of data collection.
- (ii) The text-processing sub-module implements a collection of Natu-

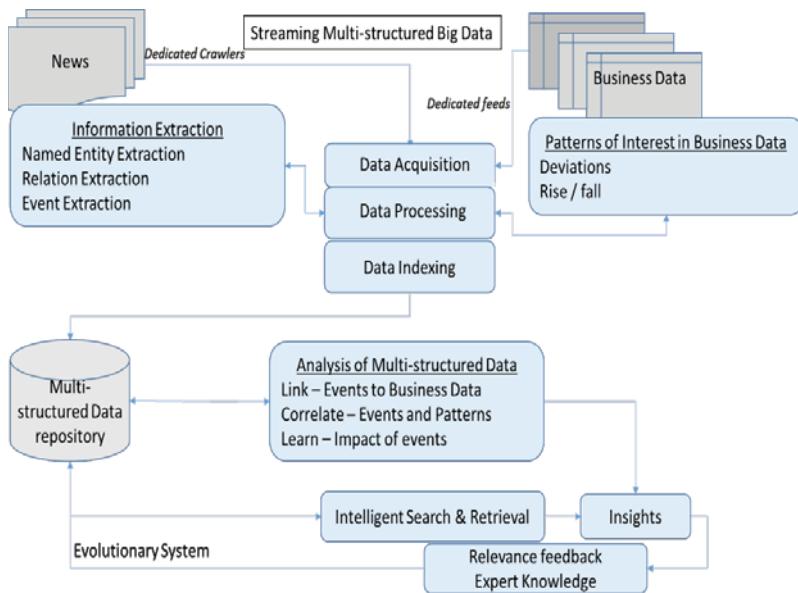


Fig. 17.3. Framework for linking structured and unstructured big data for gathering business intelligence and insights.

ral Language Processing and text mining activities that ultimately result in a set of information components that are used to index the original sources. Besides words and phrases, information components include semantically rich components like Named-entities, relations among multiple entities and events. Named entities include information components that indicate names of people, place, company, product etc. Relations are extracted from sentences using an open-source relation extractor called Ollie.* A relation is characterized by a triplet that contains subject, object and predicate. Ollie uses pre-specified patterns for sentence structures to extract these triplets. Thus all sentences may not yield relations. Events are semantically useful structured representation of key information extracted from text. An event is defined as a time-stamped information component that usually reports an “activity” about the named entities or the “actors”. Section 17.4.1 provides more details on how business-critical events can be extracted from text

*<https://knowitall.github.io/ollie/>

sources, stored and later used for search and analysis of business data. Events and their associated named entities also provide the foundations for linking the text documents to structured business data components.

- (iii) Intelligent Search and Retrieval of multi-structured data – Intelligent search facilities are provided through the use of links generated by the data processing and indexing facilities. The linking module creates links between events extracted from News documents to business data, based on overlaps among sets in terms of time of occurrence or actors. The search facilities are exploited by human analysts to view linked, and related data of different types. While the framework has inbuilt mechanisms to generate these links based on temporal co-occurrence, association rule-mining and domain rules, machine learning techniques can be deployed to learn causal associations through human guidance.
- (iv) Analytics – The analytics module employs statistical analysis and data mining to discover interesting relationships and dependencies that may exist among reported News events and business performance data. Given the vastness of the possible correlations that may emerge from data through unsupervised learning, supervised methods may be preferred more by business users. The search and retrieval module helps in providing the supervision through intelligent interactions based on evaluation of search results. Humans can also validate the automatically created links between structured and unstructured data. The system can then use these links as supervision and learn the patterns of influence. While the learnt associations do not strictly indicate causal associations, they can be validated from business users.

Different types of business intelligence applications can be built on top of this framework. These applications can be domain-dependent or domain-agnostic. For example, a generic system for analysis of World political, economic and climate News can be employed to provide early warnings about forthcoming changes in business possibilities based on learnt impacts of these types of events for business in general. Domain-specific applications could be built to analyze industry-specific News reports and generate forewarnings based on detection of potentially impactful events, after learning the impact of different types of events from past data.

The sub-sections below explain the layers in more details.

17.4.1. Extracting business-critical events from large text repositories

In this section we present the details of text processing employed to analyze large collections of business News articles to extract business-critical information from them. News articles are gathered either through RSS feeds or crawling pre-specified News web-sites using dedicated crawlers.

In the proposed framework an event is assumed to be a classified sentence that contains information of a pre-specified type. The framework can accommodate variable number of events. Business critical events may be domain-specific or domain-agnostic.

Initial supervision is provided as a labeled set of sentences, where each sentence is associated with a unique label from among a pre-specified set of categories. The system employs a k -nearest neighbor based classification algorithm to detect sentences that contain information about business events within an article and classify them [20]. The novelty lies in the sentence representation and distance computation methods that use a 500-dimensional numerical vector-based representation of words learnt from a defined corpus[†]. This can be done using the word2vec[‡] tool to compute continuous distributed representations of words [21–23]. The power of this representation lies in bringing semantically similar words closer to each other in the 500-dimensional space. The representations are not sparse. Since the repository of words is large and incremental, the word-vector store is indexed for fast retrieval of a vector for a given word during classification.

The system is trained to classify each sentence of an incoming News article based on its distance from each of a set of classified sentences in a training set. The assigned classes of the top k nearest neighbors of the sentence are then used to determine the class of the sentence based on majority voting. Experimentation yielded that a value of 7 for k yields best results for all domains.

The vector representation of each word in an incoming sentence is retrieved from the word-vector store. Distance between a pair of sentences is computed in terms of pair-wise distance among all pairs of words from them. The algorithm works as follows:

Given Training Set $T = \{\text{Set of labeled sentences where label of a sentence denotes an event class}\}$

[†]<http://www.statmt.org/lm-benchmark/1-billion-word-language-modeling-benchmark-r13output.tar.gz>

[‡]<https://code.google.com/p/word2vec/>

For each new sentence S_X extracted from incoming News articles calculate its similarity to each sentence in T as follows:

- (1) For each sentence S_T in T
 - (a) For each word w_X in S_X find the semantically closest word w_T in S_T
 - (b) Let $d_{x,t} = \text{Cosine_Similarity}(\bar{V}_X, \bar{V}_T)$, where \bar{V}_X and \bar{V}_T denote the word vectors of w_X and w_T respectively.
 - (c) Similarity of S_X and $S_T = \sum w_x d_{x,t}$, where n is the total number of words in S_X .
- (2) Select top k most similar neighbors of S_X in T .
- (3) Obtain most frequent label from the neighbors
- (4) Assign most frequent label to S_X if the label occurs more than 50% of the time among k neighbors.

Named entities present in the subject or object of event-containing sentences are considered as key “actors” associated to the event. The main verb from the predicate is further assumed to provide key information about the type of “action” or “activity” reported by the event.

A News article thereafter is characterized as a probabilistic distribution of the categories of events present in it. The events and the News articles are assigned a time-stamp that is initially same as the date-of-publication. The same event may be published in different News reports over a period of time. Sentences reporting the same event, are later grouped together to form a group which has a time-line comprised of all the publication dates in which the event was reported. The number of times a single event is reported and the total span over which the reports occur are used to compute the significance of the events. The next section explains how an event is characterized based on which events and business data are correlated and significant associations are learnt.

17.4.2. Significance of events based on their appearance in News reports

The same event may be reported in different News articles at different times using identical or similar language. A similarity-based hashing is used to group similar sentences across the repository. The following steps discuss how the grouping is done:

- (1) The word-vector of each sentence is converted to another vector using 16 Min-hash functions [24].
- (2) Projections are taken of this vector on 4 dimensional hyper-planes.
- (3) Events with same projection form a neighborhood.
- (4) A set of sentences which lie in same neighborhood and have similarity value more than 0.75 on Jaccard Similarity metric are clubbed to a group.

The group is associated with an event label, provided at least one sentence from the group is assigned a known label by the classifier in the earlier step. Min-hash practically ensures that a group usually does not get more than one label, though it is not theoretically impossible.

Let \hat{E} be the set of all unique events extracted from a repository R containing a total of M documents. Let the cardinality of \hat{E} be N . As remarked earlier, each element of \hat{E} is representative of a set of event instances, where the instances of a set can be linked back to sentences in source documents. Each event $E_i \in \hat{E}$ is assigned a unique Event-ID and contains the following derived components in its structured representation:

- (1) L_i^S = List of source documents containing the sentences
- (2) T_i = Time-line is an ordered set of time-stamps, with possible duplication constructed from the publish-times of the source documents.
- (3) $N_i = (\text{Entity-ID}|\text{Type}|\text{Frequency})$ is a list of tuples recording Named Entity occurrences in the group along with their types and frequencies.
- (4) C_i = Class of the event
- (5) τ_i = First-Report-Time of event = earliest time-stamp in T_i .
- (6) Buzz = cardinality of L_i^S
- (7) Span = Difference between the most recent time-stamp and the earliest time-stamp in T_i .

Each unique event instance for a period is finally assigned a strength based on how it was reported across various media. The significance of events thus computed, is correlated to business data for the month or the next month.

The named entities N_i and τ_i provide vital cues to link reported News events to business data. Typical examples of such data include weekly or monthly sales reports or market-share reports etc.

Figure 17.4 presents snapshots from a business News monitoring system implemented for the automobile sector using the above framework. This

Nissan	Toyota
<p>Nissan recalling 2012-14 Frontier pickups over fire risk</p> <ul style="list-style-type: none"> • First Date: Sun Oct 21, 2012 • No. of Documents: 6 • Event: Nissan recalling 2012-14 Frontier pickups over fire risk 	<p>Lab asks government to investigate Toyota Prius</p> <ul style="list-style-type: none"> • First Date: Fri Feb 15, 2013 • No. of Documents: 4 • Event: Lab asks government to investigate Toyota Prius
<p>Nissan recalls over 123k Altimas over spare tire defect - Torque News</p> <ul style="list-style-type: none"> • First Date: Tue Oct 23, 2012 • No. of Documents: 6 • Event: Nissan recalls over 123k Altimas over spare tire defect - Torque News 	<p>Gonn. books part of \$41 million settlements with Toyota and MacMillan</p> <ul style="list-style-type: none"> • First Date: Fri Feb 15, 2013 • No. of Documents: 3 • Event: Gonn. books part of \$41 million settlements with Toyota and MacMillan
<p>Nissan recall for 841000 vehicles over steering wheel problem</p> <ul style="list-style-type: none"> • First Date: Sun Oct 21, 2012 • No. of Documents: 5 • Event: Nissan recall for 841000 vehicles over steering wheel problem 	<p>Poll: Toyota Recovers US Reputation, As Chrysler & GM Lag</p> <ul style="list-style-type: none"> • First Date: Wed Feb 13, 2013 • No. of Documents: 2 • Event: Poll: Toyota Recovers US Reputation, As Chrysler & GM Lag

Fig. 17.4. Event extraction from News article for gathering competitive intelligence.

system monitors around 50 News sources that publish News related to the automobile industry across the globe. A classifier was built to identify and extract 5 different types of business events from the articles. These were legal events, future production plan related events, recall events, offers and discounts and market-news events reporting performance of stock index. Among these, the accuracy of recognizing legal events was slightly low. Other event extractions were recognized with accuracy of around 85%.

After extracting events and Named Entities, the system segregates events for the different competitors and presents information for each company grouped by events extracted from the articles. The documents are ordered by the significance of the events contained in them. Figure 17.4 shows how events can be exploited to gain competitive intelligence. The left side of the figure shows three recall events for three different models of Nissan that dominated News in October 2012. The right hand side shows legal events reporting issues faced by Toyota during February 2013. Figure 17.4 shows that interesting events are reported multiple times in many documents.

17.4.3. *Linking News events to structured business data*

The analytics module groups all event instances extracted from News articles by type and entities and generates heat-map based visualizations for human analysts. These heat maps provide competitive intelligence in terms of events. Human analysts can gain bird's eye view of an entire period through these heat-maps.

This module further links events and business data. The combined presentation of events and event heat-maps help in easier comprehension of possible correlations among events and business data. However, these are mere coincidences that need to be analyzed further. The system supports both explicit human feedback and machine-learning based validations.

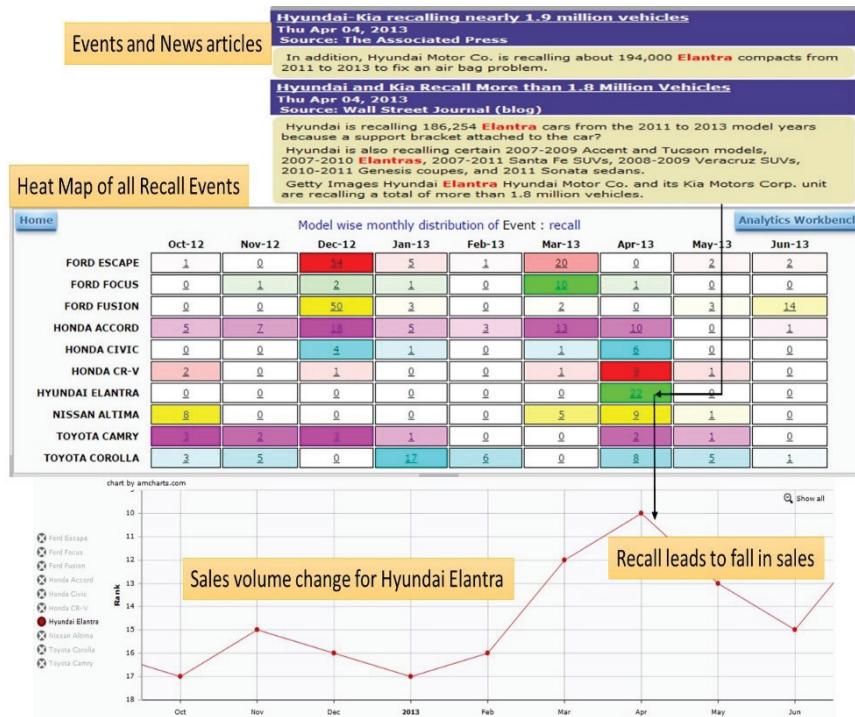


Fig. 17.5. Event consolidation shown as heatmaps and linked to structured business data. Heatmap shows large presence of recall events for Hyundai Elantra in News. The recall events are automatically linked to fall in sales of Elantra.

Figure 17.5 shows a heat-map representation that depicts all recall incidents compiled for top 10 popular automobile models of USA gathered from the News collection mentioned earlier for a nine month period from October 2012 to June 2013. The top three rows show that Ford models are often recalled and these recall events are also highly covered by the News sources. The second most widely covered recall event was that of Hyundai Elantra that occurred in April 2013. Interesting insights could be derived

when the event heat-map was aligned with the sales data for the different models. Figure 17.5 shows how drop in sales of Elantra has been linked to the recall incident.

17.4.4. Intelligent searching of News repository

While the analytics module links structured data to pre-classified business events, there may be many other types of events that could be of potential interest to business analysts. The present implementation of the framework supports intelligent searching of the News repository by entities and events and also see the retrieved News articles in conjunction with any business data of choice. Figure 17.6 presents a snapshot of a system where a business user retrieves all News articles of a specified time-period for which the stock-data is also shown. The summary box shows key events extracted, but not classified, from the News article. The user can label elements of the summary box with event labels to enhance the system with knowledge to extract and classify more events from the document collections.

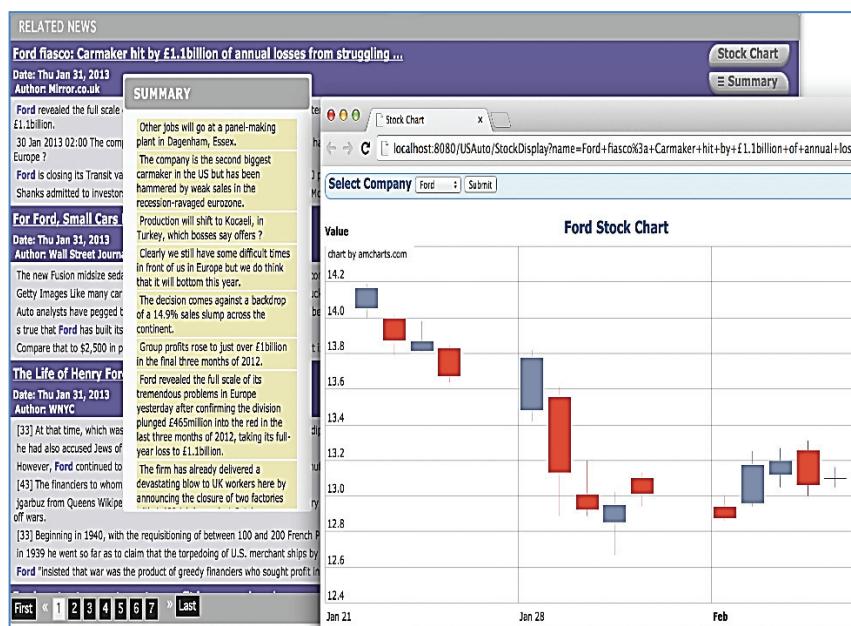


Fig. 17.6. Searching for relevant News to interpret Stock performance.

17.4.5. Learning impact of News events on business data

We now present how significant correlations can be learnt from consolidated event tables from linked data generated by the system. Table 17.2 shows a portion of a sample table built from the News collection mentioned earlier. A generic event class termed Announcements was added to the event library which accounted for all announcements by a company or with respect to a model which was not classified as one known event. This was just to see whether the presence of a company in News is related to its performance. The structured data for which event associations are learnt comprises sales figures and disclosure about the rank of a vehicle in a particular month, based on published reports by North American Dealers Association. Table 17.2 shows a partial view of event data summarized and discretized for nine-months for three competing models - Nissan Altima, Honda Accord and Toyota Camry. The event space has been discretized using a simple 2-bin discretise such that for each event type, if the number of event reports in a month is higher than its overall average, it is marked as HIGH, else LOW.

After aggregating the event data for all the popular models, we employed a Random Forest based ensemble learning method [25] to learn impact of different types of business on sales of a vehicle. Based on the published rank of a model in a month, the class-label for each month for each model was assigned No Change, Fall or Rise based on whether the rank of a model did not change, fell or rose with respect to the earlier month.

Random Forests can not only learn decision trees but can also learn to extract the impact of different types of reported events on business data recorded in structured forms. Table 17.3 presents a summary depicting impact of different types of reported events on sales data, learnt for top fifteen vehicle models sold in US. It can be concluded from Table 17.3 that the Recall event has maximum impact on its sales, followed by announcements related to new production plans or closing down an existing facility. It also shows that offers and discounts or pricing of an automobile model matters, though it has lower impact. All other routine announcements do not have much impact on sales or rank data.

Decision making process is a complex and risk-prone activity for any business. It not only requires domain expertise but is also a knowledge-driven activity, which requires awareness about all contemporary events as well as capability to analyze large volumes of heterogeneous information easily. We have presented a framework for multi-structured data analytics

Table 17.2. Events linked with Business Data for learning impact.

Company	Model	Month	Announcements	Production	Offer	Recall	Monthly_Rank
Nissan	Altima	12-Oct	LOW	LOW	LOW	HIGH	No Change
Nissan	Altima	12-Nov	LOW	LOW	LOW	LOW	Fall
Nissan	Altima	12-Dec	LOW	LOW	LOW	LOW	Gain
Nissan	Altima	13-Jan	LOW	LOW	LOW	LOW	Gain
Nissan	Altima	13-Feb	LOW	LOW	LOW	LOW	Gain
Nissan	Altima	13-Mar	LOW	LOW	LOW	HIGH	Gain
Nissan	Altima	13-Apr	HIGH	LOW	LOW	HIGH	Fall
Nissan	Altima	13-May	LOW	LOW	LOW	LOW	Gain
Nissan	Altima	13-Jun	LOW	LOW	LOW	LOW	Fall
Honda	Accord	12-Oct	HIGH	HIGH	HIGH	HIGH	Fall
Honda	Accord	12-Nov	HIGH	HIGH	HIGH	HIGH	Fall
Honda	Accord	12-Dec	HIGH	HIGH	HIGH	HIGH	No Change
Honda	Accord	13-Jan	HIGH	HIGH	HIGH	LOW	Gain
Honda	Accord	13-Feb	HIGH	HIGH	HIGH	LOW	No Change
Honda	Accord	13-Mar	HIGH	LOW	HIGH	HIGH	Fall
Honda	Accord	13-Apr	HIGH	HIGH	LOW	HIGH	Gain
Honda	Accord	13-May	HIGH	HIGH	LOW	LOW	No Change
Honda	Accord	13-Jun	LOW	LOW	LOW	LOW	Fall
Toyota	Camry	12-Oct	LOW	HIGH	LOW	LOW	Gain
Toyota	Camry	12-Nov	LOW	LOW	LOW	LOW	Fall
Toyota	Camry	12-Dec	HIGH	LOW	LOW	LOW	No Change
Toyota	Camry	13-Jan	LOW	LOW	LOW	LOW	Gain
Toyota	Camry	13-Feb	LOW	LOW	HIGH	LOW	No Change
Toyota	Camry	13-Mar	LOW	LOW	HIGH	LOW	Fall
Toyota	Camry	13-Apr	HIGH	HIGH	LOW	LOW	No Change
Toyota	Camry	13-May	HIGH	LOW	LOW	LOW	No Change
Toyota	Camry	13-Jun	LOW	LOW	LOW	LOW	Gain

and depicted the design of a system built using this framework. The system is designed to help analysts by automatically extracting and analyzing incoming News articles to extract business-critical events and link multi-structured data. The links are thereby exploited by the system to present diverse types of information together for ease of comprehension of their inter-relationships. The system has capabilities to learn the impact of different type of events on business performance data through analysis of large volumes of past data. Based on the learnt impact, an early-warning system has been generated that triggers pro-active notifications for business analysts to take a look at potentially impactful event reports as soon as they are first published. Being aware of potential impact of a reported event can help strategists in decision-making and predicting possible future scenarios ahead of competitors. Such an early-warning system provides huge competitive advantage to business leaders.

Table 17.3. Partial results for learnt impact of events on Automobile Sales Data.

Events	Impact
Recall	0.95
Production	0.89
Offer	0.36
Announcements	0.25

17.5. Conclusions

In this chapter we have considered the problem of multimodal data mining. We have indicated how we can design multimodal information integration scheme for combining text and image information. We have shown that by combining unstructured and structured data, applications can be designed for business analytics. An example system is described which help analysts by automatically extracting and analyzing incoming News articles to extract business-critical events and link multi-structured data. There are many other areas, like those involving bio-medical data and geo-physical data, in which use of multimodal data analytics can lead to meaningful insights. Deep semantic analysis exploiting dependence between heterogeneous data for extraction of actionable analytics is the key challenge in this space.

References

- [1] X. Wu, X. Zhu, G.-Q. Wu, and W. Ding. Data Mining with Big Data, *IEEE Transactions on Knowledge and Data Engineering*, **26**(1) (2014).
- [2] K. Barnard and D. A. Forsyth. Learning the semantics of words and pictures. *International Conference on Computer Vision*, II, 408–415 (2001).
- [3] K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D. M. Blei and M. I. Jordan. Matching Words and Pictures, *Journal of Machine Learning Research*, **3**, 1107–1135 (2003).
- [4] W. Shuliang, D. Gangyi, Z. Ming, Big Spatial Data Mining, *Proc. IEEE International Conference on Big Data*, 13–20 (2013).
- [5] H. J. Miller and J. Han. *Geographic Data Mining and Knowledge Discovery*. London: Taylor and Francis, 2nd edition (2009).
- [6] B. Tomaszewski. Situation awareness and virtual globes: Applications for disaster management, *Computers & Geosciences*, **37**(1), 86–92 (2011).
- [7] B. Tomaszewski. Producing geo-historical context from implicit sources: a geovisual analytics approach, *The Cartographic Journal*, **45**, 165–181 (2008).
- [8] K. Alonso and M. Datcu. Knowledge-Driven Image Mining System for Big Earth Observation Data Fusion: GIS Maps Inclusion In Active Learning Stage, *Proc. IGARSS*, 3538–3541 (2014).
- [9] C. Bhatt, M. Kankanhalli. Probabilistic Temporal Multimedia Data Mining, *ACM Trans. Intell. Syst. Technol.* **2**(2) (2011).
- [10] K. Shirahama, K. Ideno and K. Uehara. A time constrained sequential pattern mining for extracting semantic events in videos. In *Book of Multimedia Data Mining and Knowledge Discovery*, 423–446 (2008).
- [11] X. Zhu, X. Wu, A. Elmagarmid, Z. Feng and L. Wu. Video data mining semantic indexing and event detection from the association perspective. *IEEE Trans. Knowl. Data Engin.* **17**(5), 665–677 (2005).

- [12] D. M. Blei and M. I. Jordan. Modeling annotated data. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '03*, 127–134 (2003).
- [13] D. M. Blei, A. Y. Ng and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, **3**, 993–1022 (2003).
- [14] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer and R. Harshman. Indexing by latent semantic indexing. *Journal of the Society for Information Science*, **41**(6), 391–407 (1990).
- [15] E. Hassan, S. Chaudhury, and M. Gopal. Multimodal information integration for document retrieval. *Document Analysis and Recognition (ICDAR), 12th International Conference on, IEEE*, 1200–1204 (2013).
- [16] G. Heinrich. Parameter estimation for text analysis. Technical report (2004).
- [17] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 50–57 (1999).
- [18] Y. Jia, M. Salzmann and T. Darrell. Learning cross modality similarity for multinomial data. *Proceedings of the IEEE International Conference on Computer Vision*, 2407–2414 (2011).
- [19] S. Vijayanarasimhan and K. Grauman. Top-down pairwise potentials for piecing together multi-class segmentation puzzles. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, 25–32 (2010).
- [20] I. Verma, L. Dey, R. S. Srinivasan and L. Singh. Event Detection from Business News. In *Proc. PREMI 2015* (2015).
- [21] T. Mikolov, K. Chen, G. Corrado and J. Dean. Efficient Estimation of Word Representations in Vector Space, In *Proceedings of Workshop at ICLR* (2013).
- [22] T. Mikolov, I. Sutskever, K. Chen, G. Corrado and J. Dean. Distributed Representations of Words and Phrases and their Compositionality. In *Proceedings of NIPS* (2013).
- [23] T. Mikolov, W.-T. Yih and G. Zweig. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of NAACL HLT* (2013).
- [24] A. Z. Broder. On the resemblance and containment of documents, Compression and Complexity of Sequences, *Proceedings IEEE*, Positano, Amalfitan Coast, Salerno, Italy, 21–29, (1997).
- [25] L. Breiman. Random Forests. *Machine Learning* **45**(1), 5–32 (2001).

Chapter 18

Solving Classification Problems on Human Epithelial Type 2 Cells for Anti-Nuclear Antibodies Test: Traditional versus Contemporary Approaches

Arnold Wiliem and Brian C. Lovell

School of Information Technology and Electrical Engineering

The University of Queensland, Australia

a.wiliem@uq.edu.au, lovell@itee.uq.edu.au

Pathology tests play a major role in our health care system and they assist physicians to make accurate diagnoses. Recently, there has been growing interest in developing Computer-Aided Diagnostic (CAD) systems which should further improve the reliability and consistency of these tests. We discuss the Anti-Nuclear Antibodies (ANA) test via Indirect Immunofluorescence protocol on Human Epithelial Type 2 (HEp-2) cells which is normally used to determine the presence of Connective Tissue Diseases such as Sjögren's syndrome, Systemic Lupus Erythematosus and Rheumatoid Arthritis. One task in the test is to determine the pattern of positive HEp-2 cells. This chapter discusses and compares both traditional and contemporary approaches to this problem via image-based analysis. Whilst, the traditional approaches comprise several popular methods used to address generic image classification problems, the contemporary approaches apply advanced mathematical techniques to improve system accuracy. Specifically, our discussion focuses on features that reside in a non-linear statistical manifold space

18.1. Introduction

Pathology tests are critical in our healthcare system for patient diagnosis. Any inaccuracies compromises patient diagnosis and treatment. Applying image-based CAD systems is one of the solutions to leverage test consistency and reliability and also decrease test turn-around time [1–5]. For instance, Tadrous *et al.* proposed a CAD system which can be used for a rapid screening of *Ziehl-Neelsen* (ZN)-stained sections for detecting *acid-alcohol-fast bacilli* (AAFB) which cause Tuberculosis (TB) in resource-poor regional areas [1].

The Anti-Nuclear Antibody (ANA) test is commonly used by clinicians to identify the existence of Connective Tissue Diseases such as Systemic Lupus Erythematosus, Sjögren's syndrome, and Rheumatoid Arthritis [6]. The hallmark protocol for doing this is through Indirect Immunofluorescence (IIF) on Human Epithelial type 2 (HEp-2) cells [6, 7]. This is due to its high sensitivity and the large range of expression of antigens. Despite these advantages, the IIF approach is also labour intensive and time consuming [8, 9]. Each ANA specimen must be examined under a fluorescence microscope by at least two scientists. This renders the test result more subjective, and thus it has much lower reproducibility and significantly larger variabilities across personnel and laboratories [10, 11].

When combined with the subjective analysis by scientists, the results produced from CAD systems can potentially address these issues [2]. In the light of this fact, there has been a surge of interest shown in the literature to develop such systems [2, 10–28]. This also has been supported by a number of excellent benchmarking platforms such as ICPR2012Contest [2, 26], SNPHEp-2 [29] and ICIP2013^a [60] which help the community to compare methods proposed in this area. Existing approaches to image analysis have also been adapted and evaluated [16, 29].

In this chapter, we contrast two different approaches for addressing the HEp-2 cell classification problem. The first approach belongs to the traditional approaches for solving general image classification problems. Unlike the first approach, the second approach attempts to use features residing in a non-linear analytical manifold space. It has been shown in numerous works [27, 30] that modelling an image in a non-linear manifold space offers better classification performance.

We continue our discussion as follows. Section 18.2 provides background information on the ANA test. Readers who are more interested in the pattern analysis field could skip this section. As this field is still in its infancy, finding image datasets is a challenge by itself. Therefore, in Section 18.3, we discuss various efforts in the community to lay a strong foundation for the practitioners to reliably test and compare their methods. We use these datasets in order to compare the two approaches. Sections 18.4 and 18.5 discuss the both approaches. We present experimental results to compare the two approaches in Section 18.6. The chapter ends with discussions and conclusions in Section 18.7.

^aThe competition website and dataset available at <http://nerone.diiie.unisa.it/contest-icip-2013/index.shtml>

18.2. The Anti-Nuclear Antibodies Test

The ANA test is used to screen for a wide range of CTD [6, 7]. Methods to detect ANA include Indirect Immunofluorescence (IIF), Enzyme Immunosorbent Assay (EIA)/Enzyme-Linked Immunosorbent Assay (ELISA), Farr Assay, Multiplex Immunoassay (MIA) and Western Blot [31].

Amongst these methods, the IIF is considered to be the gold standard as the test detects antibodies to a wide variety of human antigens [6]. Other techniques are used as secondary/confirmatory tests. For instance, EIA/ELISA are methods specifically designed to target a single autoantigen (e.g. dsDNA, SSA-A/Ro). The Farr Assay is a radio-labeled assay for quantifying anti-dsDNA [31]. Using western blot, antigens are separated according to their molecular weight and then transferred onto a membrane or strips [31]. The strips are then incubated with the patient serum. Positive reactions are compared to a positive control strip. For MIA, serum is incubated with a suspension of multi-coloured polystyrene micro-spheres coated with a range of antigens. The binding, determining the test result, is then quantified using a specific instrument platform.

For the IIF method, once a specimen has been prepared, normally it will be examined under a fluorescent microscope by at least two scientists. The analysis starts by determining the specimen positivity from the fluorescent signal observed. The guidelines established by the Center of Disease Control and Prevention, Atlanta, Georgia (CDC) suggest the use of a scoring system ranging from 0 to 4+ wherein 0 represents negative (no fluorescent signal observed), and 4+ represents the strongest positive (very bright fluorescent signal observed) [32]. As this process is subjective, it is possible to reduce the scoring system into merely determining whether the fluorescence intensity level of the sample is positive, intermediate or negative. Then, a more objective but expensive method such as titration via serial dilution may be used. Finally, the last step in the analysis is to determine the pattern appearing in positive and intermediate specimens.

18.3. Classification Task and Benchmarking Datasets

The complete steps involved in a CAD system for ANA test is depicted in Fig. 18.1. As discussed in the previous section, the pathologists will focus on several aspects in their analysis: (1) whether the specimen is positive; (2) the fluorescent signal strength for positive specimens and (3) the positive specimen pattern. In this chapter, we confine ourselves to step

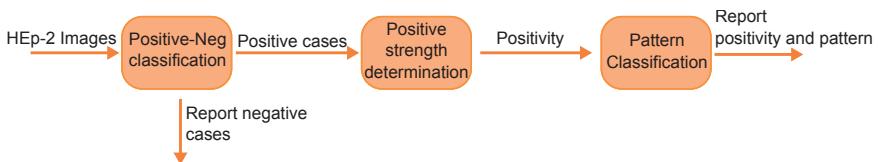


Fig. 18.1. Typical steps done in a CAD system for ANA IIF test.

3 (i.e. positive specimen pattern classification).

To determine the pattern of a positive patient, the pathologists need to analyse a number of features. Each ANA image normally contains a distribution of HEp-2 cells that can be divided into two main cell cycle stages: interphase and mitosis stages (Refer Fig. 18.2 for some example images of ANA patterns). During the mitosis stage, HEp-2 cells divide into two daughter cells. Unlike the interphase stage, in this stage the amount of cell chromatin is doubled. The cells undergoing the mitosis stage may express different antigens or antigens in different concentration to those of the interphase stage. Therefore, the pathologists will consider both the pattern of interphase and mitotic cells.

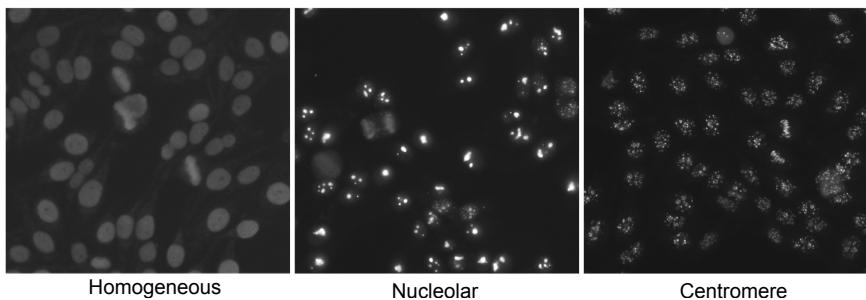


Fig. 18.2. Some examples of ANA HEp-2 images with their patterns.

From the above discussion, it is clear that the CAD systems need to determine individual cell patterns before making inference on the ANA whole image. Whilst there are some works that deal with the ANA image pattern classification, the current effort is largely concentrated on the individual cell classification [33, 34]. Specifically, most works focus on the interphase cell classification [25–28]. We note that there are only handful of works related to mitotic cell classification [35]. Henceforth, in this chapter we only focus on the interphase cell classification problem.

One of the reasons why the current effort is largely concentrated on the interphase cell classification is due to the availability of benchmark datasets. We first define the classification task for interphase cells. Several prominent benchmark datasets are discussed after that.

18.3.1. Interphase HEp-2 cell classification problem

The classification goal is to develop a classifier φ which classifies a set of HEp-2 cell images. Each image is represented by the three-tuple $(\mathbf{I}, \mathbf{M}, \delta)$ [29] : (1) \mathbf{I} represents the cell fluorescence image; (2) \mathbf{M} is the cell mask which is automatically extracted and (3) δ represents the cell positivity strength which has two values weak/borderline (*intermediate*) or strong (*positive*). Let \mathbf{Y} be a probe image, ℓ be its class label and $\mathcal{G} = \{(\mathbf{I}, \mathbf{M}, \delta)_1, \dots, (\mathbf{I}, \mathbf{M}, \delta)_n\}$ be a given gallery set. The classifier task is to predict the probe label, $\hat{\ell}$. In other words, $\varphi : \mathbf{Y} \times \mathcal{G} \mapsto \hat{\ell}$, where ideally $\hat{\ell} = \ell$.

18.3.2. Benchmark datasets

There are at least three datasets have been proposed for addressing the interphase HEp-2 cell classification problems. Although this chapter only uses two out of three datasets, for completeness, we discuss all of them.

ICPR2012 [2] - was collected for the HEp-2 Cells Classification contest held at the 21st International Conference on Pattern Recognition (ICPR 2012). The dataset comprises 1,457 cells extracted from 28 ANA images. It has six patterns: centromere, coarse speckled, cytoplasmic, fine speckled, homogeneous, and nucleolar. Each ANA image was acquired by means of a fluorescent microscope (40-fold magnification) coupled with 50W mercury vapour lamp and with a CCD camera. The cell image masks were hand labelled. The dataset can be freely obtained from <http://mivia.unisa.it/datasets/biomedical-image-datasets/hep2-image-dataset/>. Figure 18.3 presents some examples.

SNPHEp-2 [25, 29] was collected between January and February 2012 at Sullivan Nicolaides Pathology laboratory, Australia. The dataset has five patterns: centromere, coarse speckled, fine speckled, homogeneous and nucleolar. The 18-well slide of HEP-2000 assay from Immuno Concepts N.A. Ltd. with screening dilution 1:80 was used to prepare 40 ANA images. Each specimen image was captured using a monochrome high dynamic range cooled microscopy camera, which was fitted on a microscope with a plan-Apochromat 20x/0.8 objective lenses and an LED illumination source.

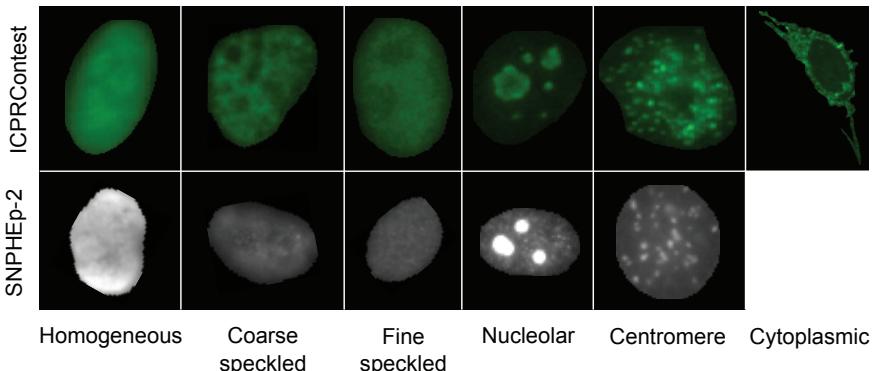


Fig. 18.3. Some examples images from the ICPR2012 [2] and SNPHEp-2 [25, 29] datasets.

DAPI image channel was used to automatically extract cell image masks. There were 1,884 cell images extracted from 40 ANA images. The ANA images were organised into training and testing sets with 20 images each (four images for each pattern). In total there were 905 and 979 cell images extracted for training and testing. The dataset can be downloaded from: <http://staff.itee.uq.edu.au/lovell/snphep2/>. Note that SNPHEp-2 is part of the larger dataset, called ICIP2013.

ICIP2013^b [60] was obtained between 2011 and 2013 at Sullivan Nicolaides Pathology laboratory, Australia. It was constructed for the Competition on Cells Classification by Fluorescent Image Analysis hosted by the 20th IEEE International Conference on Image Processing (ICIP2013). The dataset utilised 419 patient positive sera which were prepared on the 18-well slide of HEP-2000 IIF assay from Immuno Concepts N.A. Ltd. with screening dilution 1:80. The specimens were then automatically photographed using the same hardware as SNPHEp-2. Approximately 100-200 cell images were extracted from each patient serum. In total there were 68,429 cell images extracted. These were then divided into 13,596 images for training and 54,833 for testing. The dataset had six patterns: homogeneous, speckled, nucleolar, centromere, nuclear membrane and golgi. The labeling process involved at least two pathologists who read each patient specimen under a microscope. A third expert's opinion was sought to adjudicate any discrepancy between the two opinions. It used each ANA image label for the groundtruth of cells extracted from the image. Furthermore, all the labels

^b<http://nerone.diiie.unisa.it/contest-icip-2013/index.shtml>

were validated by using secondary tests such as ENA, and anti-ds-DNA to confirm the presence and/absence of specific patterns.

In this chapter we only use ICPR2012 and SNPHEp-2 datasets. This is because both datasets have similar numbers of images; making it easier to interpret the results. In addition, ICPR2012 and SNPHEp-2 datasets are acquired from two laboratories that use different assays and hardware. This means we can do robustness analysis on the methods.

18.4. Bag of Words Approach

One of the most well-known approaches for addressing general object classification problems is the bag of words approach [36, 37]. This approach utilises the local statistics of image patches in the form of histograms to represent an image. It has been shown in numerous works that image histogram representation is sufficient to address generic image classification problems [37]. We refer the readers to Zhang *et al.* [37] for a full treatment on the bag of words approach. Here we explore the efficacy of several variants of bag of word methods.

A conceptual illustration of the general approach for obtaining histograms of visual words from HEp-2 cell images is shown in Fig. 18.4. Each cell image is first resized into a canonical size and then divided into small overlapping patches. The patches are in turn represented by patch-level features. The *local* histogram from each patch is then extracted by using the pre-trained visual word dictionary. The local histograms located inside a region are pooled to compute the overall histogram for the region. Finally, the cell image is represented by a set of regional histograms; examples of regional structures are shown in Fig. 18.5.

In the following sub-sections, we first describe low-level patch-level features, followed by presenting various methods for local histogram extraction. The regional structures such as Spatial Pyramid Matching (SPM) [38], Dual Region (DR) [29] and Cell Pyramid Matching (CPM) [25] are discussed afterwards.

18.4.1. Patch-level feature extraction

Given a HEp-2 cell image ($\mathbf{I}, \mathbf{M}, \delta$), both the FITC image \mathbf{I} and mask image \mathbf{M} are divided into small overlapping patches $\mathcal{P}_I = \{\mathbf{p}_{I,1}, \mathbf{p}_{I,2}, \dots, \mathbf{p}_{I,n}\}$ and $\mathcal{P}_M = \{\mathbf{p}_{M,1}, \mathbf{p}_{M,2}, \dots, \mathbf{p}_{M,n}\}$. The division is accomplished in the same manner of both images, resulting in each patch in the FITC image

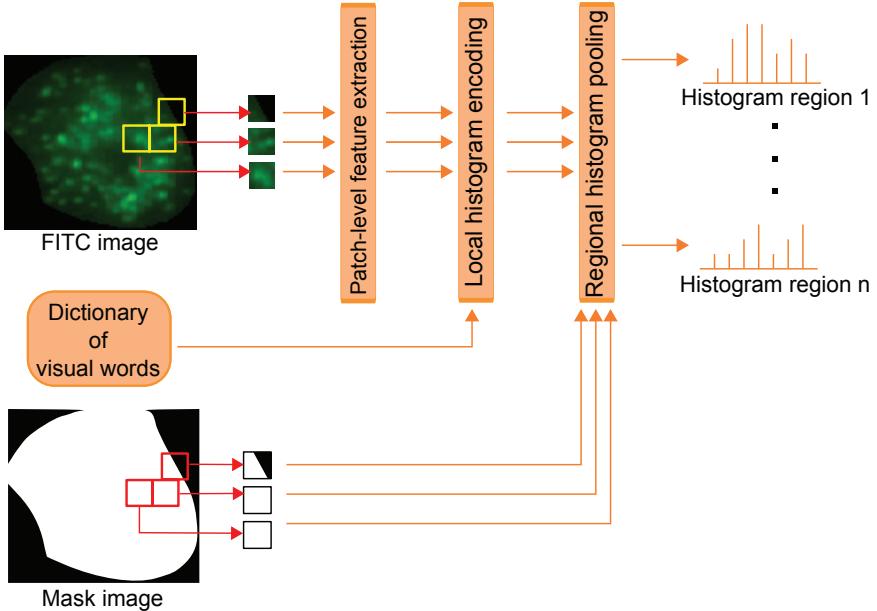


Fig. 18.4. An illustration how to obtain histograms of visual words from HEp-2 cell images.

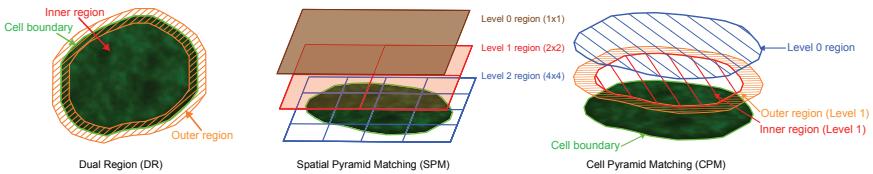


Fig. 18.5. Illustration of spatial structures: Dual Region (DR) [29]; Spatial Pyramid Matching (SPM) [38] and Cell Pyramid Matching (CPM) [25].

having a corresponding patch in the mask image. Let f be a patch-level feature extraction function $f : p_I \mapsto \mathbf{x}$, where $\mathbf{x} \in \mathbb{R}^d$. \mathcal{P}_I now can be represented as $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$.

For evaluation purposes, we selected two popular patch-level feature extraction techniques, based on the Scale Invariant Feature Transform (SIFT) and the Discrete Cosine Transform (DCT). The SIFT descriptor is invariant to uniform scaling, orientation and partially invariant to affine distortion and illumination changes [39]. These attributes are advantageous in this classification task as cell images are unaligned and have high within-class

variabilities. DCT based features proved to be effective for face recognition in video surveillance [40]. By using only the low frequency DCT coefficients (essentially a low-pass filter), each patch representation is relatively robust to small alterations [40]. We follow the extraction procedures for SIFT and DCT as per Liu *et al.* [41] and Sanderson *et al.* [40], respectively.

The dictionary of visual words, denoted \mathcal{D} , is trained from patches extracted by sliding windows from training cell images. Each histogram encoding method has a specific dictionary training procedure.

18.4.2. Generation of local histograms

For each patch-level feature that belongs to region r , $\mathbf{x}_j \in \mathcal{X}_r$, a local histogram \mathbf{h}_j is obtained. In this work we consider three prominent histogram encoding methods: (1) vector quantisation; (2) soft assignment; (3) sparse coding. The methods are elucidated below.

18.4.2.1. Vector Quantisation (VQ)

Given a set \mathcal{D} , the dictionary of visual words, the i -th dimension of local histogram \mathbf{h}_j for patch \mathbf{x}_j is computed via:

$$\mathbf{h}_{j,i} = \begin{cases} 1 & \text{if } i = \arg \min_{k \in 1, \dots, |\mathcal{D}|} \text{dist}(\mathbf{x}_j, \mathbf{d}_k) \\ 0 & \text{otherwise} \end{cases}, \quad (18.1)$$

where $\text{dist}(\mathbf{x}_j, \mathbf{d}_k)$ is a distance function between \mathbf{x}_j and \mathbf{d}_k , while \mathbf{d}_k is the k -th entry in the dictionary \mathcal{D} and $|\mathcal{D}|$ is the cardinality of \mathcal{D} . The dictionary is obtained via the k -means algorithm [42] on training patches, with the resulting cluster centers representing the entries in the dictionary.

The VQ approach is considered as a hard assignment approach since each image patch is only assigned to one of the visual words. Such hard assignment can be sensitive to noise [36].

18.4.2.2. Soft Assignment (SA)

In comparison to the VQ approach above, a more robust approach is to apply a probabilistic method [40]. Here the visual dictionary \mathcal{D} is a convex mixture of Gaussians. The i -th dimension of the local histogram for \mathbf{x}_j is calculated by:

$$\mathbf{h}_{j,i} = \frac{w_i p_i(\mathbf{x}_j)}{\sum_{k=1}^{|\mathcal{D}|} w_k p_k(\mathbf{x}_j)}, \quad (18.2)$$

where $p_i(\mathbf{x})$ is the likelihood of \mathbf{x} according to the i -th component of the visual dictionary \mathcal{D} :

$$p_i(\mathbf{x}) = \frac{\exp\left[-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^T \mathbf{C}_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i)\right]}{(2\pi)^{\frac{d}{2}} |\mathbf{C}_i|^{\frac{1}{2}}}, \quad (18.3)$$

with w_i , $\boldsymbol{\mu}_i$ and \mathbf{C}_i representing the weight, mean and diagonal covariance matrix of Gaussian i , respectively. The scalar d represents the dimensionality of \mathbf{x} . The dictionary \mathcal{D} is obtained using the Expectation Maximisation algorithm [42] on training patches.

18.4.2.3. Sparse Coding (SC)

It has been observed that each local histogram produced via Eq. (18.2) is sparse in nature (i.e. most elements are close to zero) [43]. In other words, the SA approach described in Section 18.4.2.2 is an *indirect* sparse coding approach. Hence, it is possible to adapt *direct* sparse coding algorithms in order to represent each patch as a combination of dictionary atoms [44, 45], which theoretically can lead to better recognition results [43].

A vector of weights $\boldsymbol{\vartheta} = [\vartheta_1, \vartheta_2, \dots, \vartheta_{|\mathcal{D}|}]^T$ is computed for each \mathbf{x}_j by solving a minimisation problem that selects a sparse set of dictionary atoms. As the theoretical optimality of the ℓ_1 -norm minimisation solution is guaranteed [46], in this work we use:

$$\min \frac{1}{2} \|\mathbf{D}\boldsymbol{\vartheta} - \mathbf{x}_j\|_2^2 + \lambda \sum_k \|\vartheta_k\|_1, \quad (18.4)$$

where $\|\cdot\|_p$ denotes the ℓ_p -norm and $\mathbf{D} \in \mathbb{R}^{d \times |\mathcal{D}|}$ is a matrix of dictionary atoms. The dictionary \mathbf{D} is trained by using the K-SVD algorithm [47], which is known to be suitable for obtaining reasonable dictionaries in similar cases, i.e., using a large number of small image patches [48].

As $\boldsymbol{\vartheta}$ can have negative values due to the objective function in Eq. (18.4), we construct each local histogram using the absolute value of each element in $\boldsymbol{\vartheta}$ [43]:

$$\mathbf{h}_j = [|\vartheta_1|, |\vartheta_2|, \dots, |\vartheta_{|\mathcal{D}|}|]. \quad (18.5)$$

Compared to both Eqs. (18.1) and (18.2), obtaining the histogram using sparse coding is considerably more computationally intensive, due to the need to solve a minimisation problem for each patch.

18.4.3. Histogram pooling

Let \mathcal{X}_r be the set of patch-level features belonging to region r . The overall histogram representation for region r is then obtained via averaging local histograms [40, 43]:

$$\mathbf{H}^{[r]} = \frac{1}{|\mathcal{X}_r|} \sum_{j=1}^{|\mathcal{X}_r|} \mathbf{h}_j, \quad (18.6)$$

where $|\mathcal{X}_r|$ is the number of elements in set \mathcal{X}_r . In the following subsections, we describe several possible spatial layouts for the regions and the corresponding similarity measures.

18.4.4. Spatial structures for multiple region descriptors

In this section we describe two existing spatial structures for using multiple regional descriptors (i.e. SPM and DR), followed by the CPM approach. The conceptual diagram for each approach is shown in Fig. 18.5.

18.4.4.1. Spatial Pyramid Matching (SPM)

The regions are organised similarly to an image pyramid with several levels [38]. At each level l , the image is divided into $(2^l) \times (2^l)$ non-overlapping regions. For instance, at level 0 (i.e. the top level), the image is divided into 1×1 region; at level 1, the image is divided into 2×2 regions. In this work, we follow Lazebnik *et al.* [38] by using a three-level pyramid (i.e. levels 0, 1 and 2): 1×1 , 2×2 and 4×4 . In total, there are $1 + 4 + 16 = 21$ regions. The *pyramid match kernel* is used to measure the similarities between two images [38]:

$$\mathcal{K}(\mathbf{H}_1, \mathbf{H}_2) = \frac{1}{2^L} G\left(\mathbf{H}_1^{[0,r]}, \mathbf{H}_2^{[0,r]}\right) + \sum_{l=1}^L \frac{1}{2^{L-l+1}} G\left(\mathbf{H}_1^{[l,r]}, \mathbf{H}_2^{[l,r]}\right), \quad (18.7)$$

where $\mathbf{H}_k^{[l,r]}$ is the r -th regional histogram of levels l of the k -th image, while L is the maximum number of levels (i.e. $L = 2$). $G(\cdot, \cdot)$ is a histogram intersection kernel, defined as [38]:

$$G\left(\mathbf{H}_1^{[l,r]}, \mathbf{H}_2^{[l,r]}\right) = \sum_j \min\left(\mathbf{H}_{1,j}^{[l,r]}, \mathbf{H}_{2,j}^{[l,r]}\right), \quad (18.8)$$

where $\mathbf{H}_{k,j}^{[l,r]}$ is j -th dimension of a regional histogram for level l and region r of image k .

18.4.4.2. Dual Region (DR)

Each cell is divided into an inner region, which covers the cell content, and an outer region, which contains information related to cell edges and shape [29]. To this end, each patch is first classified as either belonging to the inner or outer region by inspecting its corresponding mask patch. More specifically, let $\mathcal{X} = \mathcal{X}^{[o]} \cup \mathcal{X}^{[i]}$, with $\mathcal{X}^{[o]}$ representing the set of outer patches, and $\mathcal{X}^{[i]}$ the set of inner patches. The classification of patch \mathbf{p} into a region is done via:

$$\mathbf{p}_I \in \begin{cases} \mathcal{X}^{[o]} & \text{if } \text{fg}(\mathbf{p}_M) \in [\tau_1, \tau_2) \\ \mathcal{X}^{[i]} & \text{if } \text{fg}(\mathbf{p}_M) \in [\tau_2, 1] \end{cases}, \quad (18.9)$$

where \mathbf{p}_M is the corresponding mask patch; $\text{fg}(\mathbf{p}_M) \in [0, 1]$ computes the normalised occupation count of foreground pixels from mask patch \mathbf{p}_M ; τ_1 is the minimum foreground pixel occupation of a patch belonging to the outer region; τ_2 is the minimum pixel occupation of a patch belonging to the inner region. Note that the size of the inner and outer regions is indirectly determined via Eq. (18.9). Based on preliminary experiments, we have found that $\tau_1 = 0.3$ and $\tau_2 = 0.8$ provide good results. Unlike SPM, there are only two regional histograms required to represent a cell image. As such, the DR descriptor is $(21 - 2)/21 \approx 90\%$ smaller than SPM.

The similarity between two images is defined via:

$$\mathcal{K}(\mathbf{H}_1, \mathbf{H}_2) = \exp[-\text{dist}(\mathbf{H}_1, \mathbf{H}_2)]. \quad (18.10)$$

Adapting [40], $\text{dist}(\mathbf{H}_1, \mathbf{H}_2)$ is defined by:

$$\text{dist}(\mathbf{H}_1, \mathbf{H}_2) = \alpha^{[i]} \|\mathbf{H}_1^{[i]} - \mathbf{H}_2^{[i]}\|_1 + \alpha^{[o]} \|\mathbf{H}_1^{[o]} - \mathbf{H}_2^{[o]}\|_1, \quad (18.11)$$

where $\mathbf{H}_k^{[i]}$ and $\mathbf{H}_k^{[o]}$ are the inner and outer region histograms of image k , respectively; $\alpha^{[i]}$ and $\alpha^{[o]}$ are positive mixing parameters which define the importance of information contained for each region, under the constraint of $\alpha^{[i]} + \alpha^{[o]} = 1$. A possible drawback of the DR approach is that determining good settings for the τ_1 , τ_2 and $\alpha^{[i]}$ parameters is currently a time consuming procedure, where a heuristic or grid-based search is used. Furthermore, not all valid settings in such a search might be evaluated, which can lead to sub-optimal discrimination performance.

18.4.4.3. Cell Pyramid Matching (CPM)

The CPM approach combines the advantages of both SPM and DR structures. It adapts the idea of using a pyramid structure from SPM as well

as the inner and outer regions from DR. Unlike SPM, CPM has only two levels: level 0 which comprises the whole cell region, and level 1 which comprises of inner and outer regions. The advantages of this combination are two fold: (1) the CPM descriptor only requires 3 histograms to represent a cell image, and is hence $(21 - 3)/21 \approx 85\%$ smaller than SPM; (2) as the CPM follows the SPM construct, it employs the pyramid match kernel, which eliminates the mixing parameters in DR.

18.5. Extending Bag of Word Approach into Riemannian Manifold Space

Recently, it has been shown that representing an image as a feature residing in Riemannian manifold space, so called manifold features, offers better performance than other features. For instance, Tuzel *et al.* [30] extract d number of features from every single pixel location. The covariance matrix of these features are then calculated. In the end, each image is represented by the $d \times d$ covariance matrix. The group of covariance matrices induces a special manifold structure called the Symmetric Positive Definite (SPD) manifold, S_{++}^d . In their work, they show that the manifold features offer superior performance compared to the traditional approaches such as HOG [49] in addressing the pedestrian detection problem.

Despite its high discrimination, manifold features need different treatment. More precisely, the standard arithmetic operations in Euclidean space such as addition and multiplication are not well defined [27]. This limits the applicability of existing machineries operating in Euclidean space such as Support Vector Machine (SVM) or Linear Discriminant Analysis (LDA).

In this section we first briefly discuss the Riemannian geometries and present an extension of bag of word approach to the Riemannian manifold space.

18.5.1. Riemannian geometry

We briefly review Riemannian geometry of SPD matrices which in turn provides the grounding for techniques described in the following sections. More involved treatments on manifolds and related topics can be found in Kolár *et al.* [50], Bhatia [51] and Lui [52]. Throughout this paper we will use the following notation:

- $GL(d)$ is the general linear group, the group of real invertible $d \times d$

matrices.

- $Sym(d)$ is the space of real $d \times d$ symmetric matrices.
- \mathcal{S}_{++}^d is the space of real $d \times d$ SPD matrices.
- $Diag(\lambda_1, \lambda_2, \dots, \lambda_d)$ is a diagonal matrix formed from real values $\lambda_1, \lambda_2, \dots, \lambda_d$ on diagonal elements.
- The principal matrix logarithm $\log(\cdot)$ is defined as $\log(X) = \sum_{r=1}^{\infty} \frac{(-1)^{r-1}}{r} (\mathbf{X} - \mathbb{I})^r = \mathbf{U} \text{Diag}(\ln(\lambda_i)) \mathbf{U}^T$ with $\mathbf{X} = \mathbf{U} \text{Diag}(\lambda_i) \mathbf{U}^T$.
- The principal matrix exponential $\exp(\cdot)$ is defined as $\exp(\mathbf{X}) = \sum_{r=0}^{\infty} \frac{1}{r!} \mathbf{X}^r = \mathbf{U} \text{Diag}(\exp(\lambda_i)) \mathbf{U}^T$ with $\mathbf{X} = \mathbf{U} \text{Diag}(\lambda_i) \mathbf{U}^T$.

Before delving more into Riemannian geometry, we need to formally define the manifold features, here called **Covariance Descriptors** (CovDs). Let $f(x, y)$ be a $W \times H$ image (greyscale, color or even hyperspectral image). Let $\mathbb{O} = \{\mathbf{o}_i\}_{i=1}^n$, $\mathbf{o}_i \in \mathbb{R}^d$ be a set of observations extracted from $f(x, y)$. For example one might extract a d dimensional feature vector (such as pixel value, gradients along horizontal and vertical directions, filter responses and etc.) at each pixel, resulting in $W \times H$ observations. Then, the image $f(x, y)$ can be represented by a $d \times d$ covariance matrix of the observations as:

$$\begin{aligned} \mathbf{C}_f &= \frac{1}{n-1} \sum_{i=1}^n (\mathbf{o}_i - \mu)(\mathbf{o}_i - \mu)^T, \\ \mu &= \frac{1}{n} \sum_{i=1}^n \mathbf{o}_i. \end{aligned} \quad (18.12)$$

The entries on the diagonal of matrix \mathbf{C}_f are the variances of each feature and the nondiagonal entries are their pairwise correlations. There are several reasons as to why CovDs are attractive for representing images. Firstly, CovD provides a natural and seamless way for fusing various features. Secondly, it can reduce the impact of noisy samples in a region through the averaging operation inherently involved in its computation. Thirdly, a $d \times d$ CovD is usually low-dimensional and independent of the size of the region (it has only $\frac{1}{2}d(d+1)$ unique values), therefore it can be easily utilised to compare regions with various sizes. Fourth, CovDs can be efficiently computed using integral images (please see Tuzel *et al.* [30] for 2D integral images). Finally, affine invariant metrics exist to compare CovD which is of special significance in computer vision tasks [53].

18.5.2. Riemannian geometry of SPD matrices

The space of real $d \times d$ SPD matrices \mathcal{S}_{++}^d , forms a Lie Group, an algebraic group with a manifold structure. It is natural to use the language of Riemannian manifolds, geodesics, and all the related concepts of differential geometry when discussing \mathcal{S}_{++}^d .

A manifold is a locally Euclidean Hausdorff space whose topology has a countable base. Locally Euclidean just means that each point has some neighbourhood that is homeomorphic to an open ball in \mathbb{R}^N for some N . Moreover, being a Hausdorff space means that distinct points have disjoint neighbourhoods. This property is useful for establishing the notion of a differential manifold, as it guarantees that convergent sequences have a single limit point.

A natural way to measure nearness on a manifold is by considering the geodesic distance between points. The geodesic distance between two points on the manifold is defined as the length of the shortest curve connecting the two points. Such shortest curves are known as geodesics and are analogous to straight lines in \mathbb{R}^N . The tangent space at a point \mathbf{P} on the manifold, $T_{\mathbf{P}}\mathcal{M}$, is a vector space that consists of the tangent vectors of all possible curves passing through \mathbf{P} . Two operators, namely the exponential map $\exp_{\mathbf{P}}(\cdot) : T_{\mathbf{P}}\mathcal{M} \rightarrow \mathcal{M}$ and the logarithm map $\log_{\mathbf{P}}(\cdot) = \exp_{\mathbf{P}}^{-1}(\cdot) : \mathcal{M} \rightarrow T_{\mathbf{P}}\mathcal{M}$, are defined over differentiable manifolds to switch between the manifold and tangent space at \mathbf{P} (see Fig. 18.6 for a conceptual illustration). The exponential operator maps a tangent vector Δ to a point \mathbf{X} on the manifold. The property of the exponential map ensures that the length of Δ is equivalent to the geodesic distance between \mathbf{X} and \mathbf{P} . The logarithm map is the inverse of the exponential map and maps a point on the manifold to the tangent space $T_{\mathbf{P}}$. The exponential and logarithm maps vary as point \mathbf{P} moves along the manifold.

The Affine Invariant Riemannian Metric (AIRM) [53]) defined as $\langle \mathbf{v}, \mathbf{w} \rangle_{\mathbf{P}} := \langle \mathbf{P}^{-1/2}\mathbf{v}\mathbf{P}^{-1/2}, \mathbf{P}^{-1/2}\mathbf{w}\mathbf{P}^{-1/2} \rangle = \text{tr}(\mathbf{P}^{-1}\mathbf{v}\mathbf{P}^{-1}\mathbf{w})$ for $\mathbf{P} \in \mathcal{S}_{++}^d$ and $\mathbf{v}, \mathbf{w} \in T_{\mathbf{P}}\mathcal{M}$, induces the following geodesic distance between points \mathbf{X} and \mathbf{Y} :

$$\delta_R(\mathbf{X}, \mathbf{Y}) = \|\log(\mathbf{X}^{-1/2}\mathbf{Y}\mathbf{X}^{-1/2})\|_F. \quad (18.13)$$

Amongst several interesting properties (see for example [51, 53]), the invariance to affine transforms, i.e. $\forall \mathbf{A} \in GL(d)$, $\delta_R^2(\mathbf{X}, \mathbf{Y}) = \delta_R^2(\mathbf{A}\mathbf{X}\mathbf{A}^T, \mathbf{A}\mathbf{Y}\mathbf{A}^T)$, is especially attractive to the computer vision community. For the AIRM, the exponential and logarithms maps are given

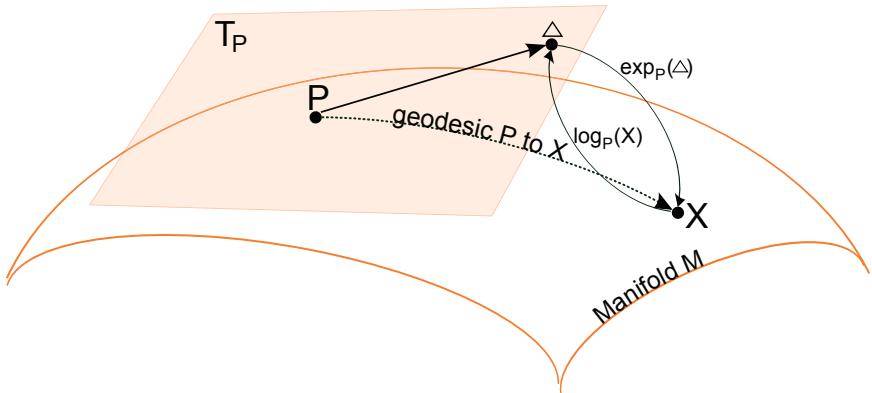


Fig. 18.6. Illustration of the tangent space T_P at point P on a Riemannian manifold \mathcal{M} . The tangent vector Δ can be obtained through the logarithm mapping, i.e. $\Delta = \log_P(X)$. Every tangent vector in T_P can be mapped back to the manifold through the exponential map, i.e. $\exp_P(\Delta) = X$. The dotted line shows the geodesic starting at P and ending at X .

by:

$$\exp_P(X) = P^{\frac{1}{2}} \exp(P^{-\frac{1}{2}} X P^{-\frac{1}{2}}) P^{\frac{1}{2}}, \quad (18.14)$$

$$\log_P(X) = P^{\frac{1}{2}} \log(P^{-\frac{1}{2}} X P^{-\frac{1}{2}}) P^{\frac{1}{2}}. \quad (18.15)$$

18.5.3. Bag of Riemannian Words (BoRW)

In this section we discuss how a conventional BoW model can be extended to incorporate Riemannian structure of manifold features. While our treatment here is directed towards the geometry of SPD matrices, concepts from this section can be easily generalised to any type of Riemannian manifolds. To devise a BoRW model, we should address two sub-problems;

- (1) Given a set of points on a SPD manifold, how can a Riemannian dictionary can trained?
- (2) Given a Riemannian dictionary, how can an image histogram be obtained for a query image?

18.5.4. Riemannian dictionary

We note that SPD matrices form a closed set under normal matrix addition, i.e. adding two SPD matrices results in another SPD matrix. Therefore, in

the simplest and most straightforward form, one could neglect the Riemannian geometry of SPD matrices and vectorise SPD matrices by extracting upper-triangular (or equivalently lower-triangular) parts of them. Having data vectorised, then a dictionary can be generated by applying k-means algorithm over vectorised data. As a result, for each cluster, the cluster centre is given by the arithmetic mean of nearest training samples to that cluster. However, the arithmetic mean is not adequate in many situations, for two main reasons. Firstly, symmetric matrices with negative or zero eigenvalues are at a finite distance from any SPD matrix in this framework. In many problems like diffusion tensor MRI, this is not physically acceptable [53, 54]. Secondly, in our application, an SPD matrix corresponds to a covariance matrix. The value of the determinant of a covariance matrix is a measure of the dispersion of the associated multivariate Gaussian. As shown for example by Pennec *et al.* the arithmetic mean of SPD matrices often leads to a swelling effect [53], that is the determinant of arithmetic mean could become greater than samples' determinants which is very undesirable [54]. As such, we seek possibilities to take into account the geometry of SPD matrices in designing a dictionary.

Let $\mathbb{X} = \{\mathbf{X}_i\}_{i=1}^N$ be a set of training samples from the underlying SPD manifold. A dictionary $\mathbb{D} = \{\mathbf{D}_i\}_{i=1}^k$, $\mathbf{D}_i \in \mathcal{S}_{++}^d$ on \mathcal{S}_{++}^d can be obtained through intrinsic k-means algorithm [53]. The core ingredient of intrinsic k-means algorithm is the Karcher mean [53]. Given an abstract manifold \mathcal{M} with associated geodesic distance $d_g(\cdot, \cdot)$, the classical generalization of the Euclidean mean for a set of points $\{\mathbf{X}_i\}_{i=1}^m$, $\mathbf{X}_i \in \mathcal{M}$ is given by the Karcher mean (also referred as Fréchet or Riemannian mean) as the point minimizing the following metric dispersion:

$$\mathbf{X}^* = \arg \min_{\mathbf{X}} \sum_{i=1}^m d_g^2(\mathbf{X}_i, \mathbf{X}). \quad (18.16)$$

Since the manifold of SPD matrices has a non-positive curvature, (18.16) has just one and only one optimum [51]. Moreover, at the optimum the gradient is zero and therefore an intrinsic gradient descent algorithm can be utilised to obtain the Karcher mean [53]. The details of computing the Karcher mean over SPD manifolds are given in Algorithm 18.1.

Having an intrinsic method for computing mean at our disposal, we seek to estimate k clusters C_1, C_2, \dots, C_k with centres $\{\mathbf{D}_i\}_{i=1}^k$ such that the sum of square geodesic-distances, i.e. $\sum_{i=1}^k \sum_{\mathbf{X}_j \in C_i} d_g^2(\mathbf{X}_j, \mathbf{D}_i)$, over all clusters is minimized. Similar to standard k-means, this can be solved using an EM-based approach. The algorithm starts by selecting k points

Algorithm 18.1 Karcher mean algorithm over \mathcal{S}_{++}^d

Input:

- A set of k points $\{\mathbf{X}_i\}_{i=1}^m$ on the underlying \mathcal{S}_{++}^d manifold,
- $maxIter$, maximum number of iterations

Output:

- The sample Karcher mean μ

- 1: Let $\mu^{(0)}$ be an initial estimate of the Karcher mean, for example by selecting a sample from $\{\mathbf{X}_i\}_{i=1}^m$ randomly.
- 2: **while** $t < maxIter$ **do**
- 3: For each point \mathbf{X}_i , compute the tangent vector \mathbf{v}_i on the current estimate of the Karcher mean ,i.e., $\mathbf{v}_i = \log_{\mu^{(t)}}(\mathbf{X}_i)$.
- 4: Compute the average tangent vector $\bar{\mathbf{v}} = \frac{1}{k} \sum_{i=1}^m \mathbf{v}_i$.
- 5: **if** $\|\bar{\mathbf{v}}\|_2$ is small **then**
- 6: $\mu = \exp_{\mu^{(t)}}(\bar{\mathbf{v}})$
- 7: **break** the while loop.
- 8: **else**
- 9: compute the new estimate of Karcher mean $\mu^{(t+1)}$ by moving along the average tangent direction ,i.e., $\mu^{(t+1)} = \exp_{\mu^{(t)}}(\epsilon \bar{\mathbf{v}})$, where $\epsilon > 0$ is small step size.
- 10: **end if**
- 11: $t \leftarrow t + 1$
- 12: **end while**

from \mathbb{X} randomly as the cluster centres. In the E-step, we assign each of the points of the dataset to the nearest cluster centre. Then in the M-step, the cluster centres are recomputed using the Karcher mean. The procedure is summarized in Algorithm 18.2.

Algorithm 18.2 Intrinsic k-means algorithm over \mathcal{S}_{++}^d for learning the visual dictionary

Input:

- training set $\mathbb{X} = \{\mathbf{X}_i\}_{i=1}^N$ from the underlying \mathcal{S}_{++}^d manifold,
- $nIter$, the number of iterations

Output:

- Visual dictionary $\mathbb{D} = \{\mathbf{D}_i\}_{i=1}^k, \mathbf{D}_i \in \mathcal{S}_{++}^d$

- 1: Initialise the dictionary $\mathbb{D} = \{\mathbf{D}_i\}_{i=1}^k$ by selecting N samples from \mathbb{X} randomly
- 2: **for** $t = 1 \rightarrow nIter$ **do**
- 3: Assign each point \mathbf{X}_i to its nearest cluster in \mathbb{D} by computing $\delta_R(\mathbf{X}_i, \mathbf{D}_j) = \|\log(\mathbf{X}_i^{-1/2} \mathbf{D}_j \mathbf{X}_i^{-1/2})\|_F$, $1 \leq i \leq N, 1 \leq j \leq k$
- 4: Recompute cluster centres $\{\mathbf{D}_i\}_{i=1}^k$ by Karcher mean algorithm
- 5: **end for**

18.5.5. Generation of image histograms

In its most straightforward and simplest form, for a set of local CovD, $\mathbb{Q} = \{\mathbf{Q}_i\}_{i=1}^p$, extracted from a query image, a signature is obtained by hard-assignment (Vector Quantisation). This demands $p \times k$ comparisons which could be prohibitive for large dictionaries given the high computational cost of AIRM. Recently, the symmetric Stein divergence is proposed as a metric on \mathcal{S}_{++}^d [55]. This metric is defined as:

$$S(\mathbf{X}, \mathbf{Y}) \triangleq \ln \left| \frac{\mathbf{X} + \mathbf{Y}}{2} \right| - \frac{1}{2} \ln |\mathbf{XY}|, \text{ for } \mathbf{X}, \mathbf{Y} \succ 0, \quad (18.17)$$

where $|\cdot|$ denotes determinant. Similar to AIRM, the symmetric Stein divergence is invariant to affine transform [55]. Other important properties of symmetric Stein divergence are summarised below.

Property 1. Let $\mathbf{X}, \mathbf{Y} \in \mathcal{S}_{++}^d$, and $\delta_T(\mathbf{X}, \mathbf{Y}) = \max_{1 \leq i \leq d} \{|\log \Lambda(\mathbf{XY}^{-1})|\}$ be the Thompson metric [56] with $\Lambda(\mathbf{XY}^{-1})$ representing the vector of eigenvalues of \mathbf{XY}^{-1} . The following sandwiching inequality between the symmetric Stein divergence and Riemannian metric exists [55]:

$$S(\mathbf{X}, \mathbf{Y}) \leq \frac{1}{8} \delta_R^2(\mathbf{X}, \mathbf{Y}) \leq \frac{1}{4} \delta_T(\mathbf{X}, \mathbf{Y}) (S(\mathbf{X}, \mathbf{Y}) + d \log d). \quad (18.18)$$

Property 2. The curve $\gamma(p) \triangleq \mathbf{X}^{\frac{1}{2}} \left(\mathbf{X}^{-\frac{1}{2}} \mathbf{Y} \mathbf{X}^{-\frac{1}{2}} \right)^p \mathbf{X}^{\frac{1}{2}}$ parameterises the unique geodesic between the SPD matrices \mathbf{X} and \mathbf{Y} . On this curve the Riemannian geodesic distance satisfies $\delta_R(\mathbf{X}, \gamma(p)) = p \delta_R(\mathbf{X}, \mathbf{Y})$; $p \in [0, 1]$ [51]. The symmetric Stein divergence satisfies a similar but slightly weaker result, $S(\mathbf{X}, \gamma(p)) \leq p S(\mathbf{X}, \mathbf{Y})$.

The first property establishes a bound between the AIRM and Stein divergence, providing the motivation for addressing Riemannian problems via the divergence. The second property reinforces the motivation by explaining that the behaviour of Stein divergences along geodesic curves is similar to true AIRM metric. In this work, we use the Stein metric instead of AIRM for computing the signatures for BoRW. The computational burden of AIRM as compared to Stein metric has been recently studied by Cherian *et al.* [57].

18.6. Results

Here we perform experiments on two datasets: ICPR2012 [2] and SNPHEp-2 [25, 29]. We exclude the ICIP2013 dataset for a number reasons. Firstly,

both ICPR2012 and SNPHEp-2 datasets are acquired from two pathology laboratories with different hardware and assays. This facilitates robustness analysis. Secondly, ICIP2013 dataset is significantly larger than these two datasets. Thirdly, both ICIP2013 and SNPHEp-2 datasets are acquired from the same laboratory. In fact, SNPHEp-2 is a part of ICIP2013.

We first discuss the experimental setup. Then, we present the results.

18.6.1. Experimental setup

We first compare the six variant of the BoW descriptors, where each of the two low-level feature extraction techniques (SIFT and DCT) is coupled with three possible methods for generating the histograms of visual words (VQ, SA and SC). The six variants are used within the framework of DR, SPM and CPM spatial structures. More precisely, each variant is denoted by *[patch-level features]-[histogram encoding method]*. For instance the variant using DCT as its patch-level features and VQ as its encoding method is called DCT-VQ. We then pick the optimal system and contrast it with the BoRW approach. To reduce the number of variations we only use the VQ method to generate BoRW histogram.

More precisely, from each cell image a set of CovDs is extracted and then vector quantised to generate image signatures. To generate CovDs from a cell image, a feature vector is assigned to each pixel in the image through using a Gabor filter-bank. More specifically, each pixel is described by

$$F_{x,y} = \left[I(x,y), x, y, |G_{0,0}(x,y)|, |G_{0,1}(x,y)|, \dots, |G_{0,v}(x,y)|, |G_{1,0}(x,y)|, \dots, |G_{u,v}(x,y)| \right], \quad (18.19)$$

where $I(x,y)$ is the intensity value at position x,y and $G_{u,v}(x,y)$ is the response of a 2D Gabor wavelet [58] centred at x,y with orientation u and scale v :

$$G_{u,v}(x,y) = \frac{k_v^2}{4\pi^2} \sum_{t,s} e^{-\frac{k_v^2}{8\pi^2}((x-s)^2+(y-t)^2)} \left(e^{ik_v((x-t)\cos(\theta_u)+(y-s)\sin(\theta_u))} - e^{-2\pi^2} \right), \quad (18.20)$$

with $k_v = \frac{1}{\sqrt{2^{v-1}}}$ and $\theta_u = \frac{\pi u}{8}$. Then, a set of overlapping blocks are extracted from the cell image and the CovD for each block is obtained. We emphasize that only foreground pixels will be considered in computing CovD of each block.

18.6.2. Combinations of local features, histogram generation and spatial structures for bag of word approach

The results, presented in Table 18.1, indicate that in most cases the proposed CPM system obtains the best performance, suggesting that it is taking advantage of both the specialised spatial layout for cells inherited from the DR approach, and the pyramid match kernel inherited from the SPM approach. The results also show that in most cases the use of DCT based patch-level feature extraction leads to better performance than using SIFT based feature extraction. We conjecture that DCT obtains better performance as the SIFT descriptor needs a larger spatial support and is hence more likely to be affected by image deformations. Specifically, SIFT divides a given image patch into 4×4 subregions in which each has 4×4 bins, followed by extracting gradient information from each subregion [39]. Therefore, SIFT needs a spatial support of at least 16×16 pixels, which is relatively large when compared to the canonical cell image size of 64×64 . In contrast, standard DCT requires a much smaller spatial support of 8×8 pixels, making it less susceptible to image deformations.

Table 18.1. Performance comparison of BoW descriptor variants on the ICPR2012 and SNPHEp-2 datasets, using various spatial configurations (DR, SPM, CPM). The scores for SNPHEp-2 dataset shown as average correct classification rate. DR = dual region; SPM = Spatial Matching Pyramid; CPM = Cell Pyramid Matching.

Descriptor Variant	ICPR2012			SNPHEp-2		
	DR	SPM	CPM	DR	SPM	CPM
DCT-SA	64.9	64.3	65.9	79.5	80.3	81.2
DCT-VQ	54.5	57.1	61.2	80.7	77.9	80.8
DCT-SC	52.6	57.9	57.2	71.0	70.5	73.5
SIFT-SA	51.6	57.5	47.8	71.6	69.7	73.2
SIFT-VQ	55.6	53.8	59.0	64.9	74.4	75.0
SIFT-SC	60.8	59.9	62.1	76.2	73.6	76.3

18.6.3. Contrasting between BoW and BoRW

In this part we contrast between the optimal Bag of Word (BoW) variant (i.e. DCT-SA with CPM) to the Bag of Riemannian Word (BoRW).

Table 18.2 shows that whilst BoRW approach outperforms the BoW in ICPR2012, it has significantly worse performance in SNPHEp-2. The

Table 18.2. Performance comparison between BoW and BoRW on the ICPR2012 and SNPHEp-2 datasets. The scores for SNPHEp-2 dataset shown as average correct classification rate. DR = dual region; SPM = Spatial Matching Pyramid; CPM = Cell Pyramid Matching.

Descriptor	ICPR2012	SNPHEp-2	average
DCT-SA + CPM	65.9	81.2	73.55
BoRW	68.1	67.6	67.85

significant performance gains by BoW could be due to the unique spatial structure specifically designed for this purpose [29]. We note that even without this spatial structure the BoRW approach could outperform the BoW approach in the ICPR2012; indicating the efficacy of the manifold features.

To further analyse the robustness of BoW and BoRW, we design a more challenging experiment protocol. More precisely, for both approaches, the dictionaries are trained on the training set of either ICPR2012 or SNPHEp-2 datasets and then used to extract the histograms in the other dataset and referred to as “ICPR2012 → SNPHEp-2” and “SNPHEp-2 → ICPR2012”, respectively. The last two tests can be understood as domain shift problem [59].

The results for the three aforementioned experiments are shown in Table 18.3. The BoRW method gives a decent performance in all the experiments. A notable example is the “ICPR2012 → SNPHEp-2” where the difference between the BoRW and BoW exceeds 18 percentage points. This indicates that the BoRW method is more robust for condition when the training images are different to the test images. There are a number of possible scenarios which will get greatly benefit from this. One notable example is when physically transporting specimens from one laboratory to the others may not be possible due to geographical limitations. Thus, the most effective way is to send the scanned images via the internet. In this case, it is desirable that the sender would only send the images of interest.

Table 18.3. Cell correct classification rate (in %) comparisons between the proposed BoRW and FT approaches against the state-of-the-art methods on inter laboratory experiments.

Method	ICPR + SNPHEp-2	ICPR → SNPHEp-2	SNPHEp-2 → ICPR
BoW	65.6	41.3	49.0
BoRW	63.6	59.7	64.9

This leaves the recipient laboratory to use its own image collections for training the system. As these collections may be generated from different process to the the images from the sender, the system needs to have the ability to deal with the domain shift problem.

18.7. Discussion and conclusion

The Anti-Nuclear Antibody (ANA) test via the Indirect Immunofluorescence protocol is considered as the hallmark protocol for identifying the existence of Connective Tissue Diseases. Unfortunately, despite its advantages, the method is labour intensive and time consuming. The use of Computer Aided Diagnosis (CAD) systems could be used to address these issues.

In this chapter, we contrasted two approaches for addressing the interphase HEp-2 cell classification problem. These approaches represent the traditional approach and the contemporary approach.

The traditional approach is represented by the Bag of Word (BoW) method. This method has been shown to be effective to address general image classification problems. We explored several variants of BoW in this application and found that it achieved reasonable performance. The best variant was to use Discrete Cosines Transform (DCT) coefficients to represent the patch-level features in conjunction with the Soft Assignment. This variant was also found to be effective for addressing the face recognition problem in the non-cooperative video surveillance domain. Furthermore, we found meaningful performance gains by applying a spatial structure, called Cell Pyramid Matching, specifically designed for this task.

In the contemporary approach, the image patches are represented as manifold features; features that lie on a lower dimensional manifold embedded in the higher dimensional space. The manifold features are opted as they have been shown to be more superior than the traditional features in various classification tasks. Unfortunately, as these features lie in a manifold space, then one needs to develop a new set of tools.

In this chapter, we presented the extension of BoW approach, here called the Bag of Riemannian Word (BoRW). The BoRW first extracts the manifold features from each image patch. Then, the manifold features are clustered using the intrinsic clustering method, called Karcher mean. The image histogram is then calculated by using Vector Quantisation.

Both the BoW and BoRW have their own advantages and weaknesses. For instance, when optimised BoW significantly outperforms the BoRW.

In the other hand, BoRW achieves superior performance than the BoW in more difficult cases when the training and testing images are generated from different sources. We note that the performance of BoRW has the potential to surpass the BoW performance. This evidence can be seen from the fact that BoRW is able to outperform BoW in ICPR2012. Currently BoRW does not use more advanced encoding method such as Soft Assignment and it also does not impose any spatial structure.

From these observations, the following conclusions could be drawn: (1) when optimised, the traditional approach, BoW, is sufficient for scenarios within a single laboratory; (2) the contemporary approach, BoRW, is more suitable to address scenarios when multiple laboratories are sharing their scanned images and (3) BoRW has potential to replace the current traditional approach in the future (e.g. by adapting the spatial structure into the model).

18.8. Acknowledgement

This work was funded by the Australian Research Council (ARC) Linkage Projects Grant LP130100230 and Sullivan Nicolaides Pathology, Australia. Arnold Wiliem is funded by the Advance Queensland Early-career Research Fellowship.

References

- [1] P. J. Tadrous, Computer-assisted screening of ziehl-neelsen-stained tissue for mycobacteria. algorithm design and preliminary studies on 2,000 images, *American Journal of Clinical Pathology*. **133**(6), 849–858 (2010).
- [2] P. Foggia, G. Percannella, P. Soda, and M. Vento, Benchmarking hep-2 cells classification methods, *IEEE transactions on medical imaging*. **32**(10) (2013).
- [3] R. D. Labati, V. Piuri, and F. Scotti. All-IDB: the acute lymphoblastic leukemia image database for image processing. In *2011 18th IEEE International Conference on Image Processing (ICIP)*, pp. 2045–2048, IEEE (2011).
- [4] D. C. Wilbur, Digital cytology: current state of the art and prospects for the future, *Acta Cytologica*. **55**(3), 227–238 (2011).
- [5] M. N. Gurcan, L. E. Boucheron, A. Can, A. Madabhushi, N. M. Rajpoot, and B. Yener, Histopathological image analysis: A review, *Biomedical Engineering, IEEE Reviews in*. **2**, 147–171 (2009).
- [6] P. L. Meroni and P. H. Schur, ANA screening: an old test with new recommendations, *Annals of the Rheumatic Diseases*. **69**(8), 1420 –1422 (2010).
- [7] A. S. Wiik, M. Hier-Madsen, J. Forslid, P. Charles, and J. Meyrowitsch,

- Antinuclear antibodies: A contemporary nomenclature using HEp-2 cells, *Journal of Autoimmunity*. **35**, 276 – 290 (2010).
- [8] N. Bizzaro, R. Tozzoli, E. Tonutti, A. Piazza, F. Manoni, A. Ghirardello, D. Bassetti, D. Villalta, M. Pradella, and P. Rizzotti, Variability between methods to determine ANA, anti-dsDNA and anti-ENA autoantibodies: a collaborative study with the biomedical industry, *Journal of Immunological Methods*. **219**(1-2), 99–107 (1998).
 - [9] B. Pham, S. Albareda, A. Guyard, E. Burg, and P. Maisonneuve, Impact of external quality assessment on antinuclear antibody detection performance, *Lupus*. **14**(2), 113–119 (2005).
 - [10] R. Hiemann, T. Bttner, T. Krieger, D. Roggenbuck, U. Sack, and K. Conrad, Challenges of automated screening and differentiation of non-organ specific autoantibodies on HEp-2 cells, *Autoimmunity Reviews*. **9**(1), 17–22 (2009).
 - [11] P. Soda and G. Iannello, Aggregation of classifiers for staining pattern recognition in antinuclear autoantibodies analysis, *IEEE Trans. Information Technology in Biomedicine*. **13**(3), 322–329 (2009).
 - [12] P. Elbischger, S. Geerts, K. Sander, G. Zervogel-Lukas, and P. Sinah, Algorithmic framework for HEp-2 fluorescence pattern classification to aid autoimmune diseases diagnosis. In *IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pp. 562–565 (2009).
 - [13] T. Hsieh, Y. Huang, C. Chung, and Y. Huang, HEp-2 cell classification in indirect immunofluorescence images. In *International Conference on Information, Communications and Signal Processing*, pp. 1–4 (2009).
 - [14] E. Cordelli and P. Soda, Color to grayscale staining pattern representation in IIF. In *International Symposium on Computer-Based Medical Systems*, pp. 1–6 (2011).
 - [15] A. Wiliem, P. Hobson, R. Minchin, and B. Lovell, An automatic image based single dilution method for end point titre quantitation of antinuclear antibodies tests using HEp-2 cells. In *Digital Image Computing: Techniques and Applications*, pp. 1–6 (2011).
 - [16] P. Strandmark, J. Ulén, and F. Kahl, Hep-2 staining pattern classification. In *International Conference on Pattern Recognition (ICPR)*, pp. 33–36 (2012).
 - [17] W. Bel Haj Ali, P. Piro, D. Giampaglia, T. Pourcher, and M. Barlaud, Biological cell classification using bio-inspired descriptor in a boosting k-NN framework. In *International Conference on Pattern Recognition (ICPR)*, pp. 1–6 (2012).
 - [18] I. Theodorakopoulos, D. Kastaniotis, G. Economou, and S. Fotopoulos, HEp-2 cells classification via fusion of morphological and textural features. In *IEEE International Conference on Bioinformatics and Bioengineering (BIBE)*, pp. 689–694 (2012).
 - [19] G. Thibault and J. Angulo, Efficient statistical/morphological cell texture characterization and classification. In *International Conference on Pattern Recognition (ICPR)*, pp. 2440–2443 (2012).
 - [20] S. Ghosh and V. Chaudhary, Feature analysis for automatic classification of hep-2 florescence patterns: Computer-aided diagnosis of auto-immune diseases. In *International Conference on Pattern Recognition (ICPR)*, pp.

- 174–177 (2012).
- [21] K. Li and J. Yin. Multiclass boosting svm using different texture features in hep-2 cell staining pattern classification. In *International Conference on Pattern Recognition (ICPR)*, pp. 170–173 (2012).
 - [22] S. D. Cataldo, A. Bottino, E. Ficarra, and E. Macii. Applying textural features to the classification of hep-2 cell patterns in iif images. In *International Conference on Pattern Recognition (ICPR)* (2012).
 - [23] V. Snell, W. Christmas, and J. Kittler. Texture and shape in fluorescence pattern identification for auto-immune disease diagnosis. In *International Conference on Pattern Recognition (ICPR)*, pp. 33750–33753 (2012).
 - [24] I. Ersoy, F. Bunyak, J. Peng, and K. Palaniappan. HEp-2 cell classification in IIF images using shareboost. In *International Conference on Pattern Recognition (ICPR)*, pp. 3362–3365 (2012).
 - [25] A. Wiliem, C. Sanderson, Y. Wong, P. Hobson, R. F. Minchin, and B. C. Lovell, Automatic classification of human epithelial type 2 cell indirect immunofluorescence images using cell pyramid matching, *Pattern Recognition*. **47**(7), 2315–2324 (2014).
 - [26] P. Foggia, G. Percannella, A. Saggesse, and M. Vento, Pattern recognition in stained hep-2 cells: Where are we now?, *Pattern Recognition*. **47**, 2305–2314 (2014).
 - [27] M. Faraki, M. T. Harandi, A. Wiliem, and B. C. Lovell, Fisher tensors for classifying human epithelial cells, *Pattern Recognition*. **47**(7), 2348–2359 (2014).
 - [28] Y. Yang, A. Wiliem, A. Alavi, B. C. Lovell, and P. Hobson, Visual learning and classification of human epithelial type 2 cell images through spontaneous activity patterns, *Pattern Recognition*. **47**(7), 2325–2337 (2014).
 - [29] A. Wiliem, Y. Wong, C. Sanderson, P. Hobson, S. Chen, and B. C. Lovell. Classification of human epithelial type 2 cell indirect immunofluorescence images via codebook based descriptors. In *IEEE Workshop on Applications of Computer Vision (WACV)*, pp. 95–102 (2013).
 - [30] O. Tuzel, F. Porikli, and P. Meer, Pedestrian detection via classification on riemannian manifolds, *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*. **30**(10), 1713–1722 (2008).
 - [31] Y. Kumar, A. Bhatia, and R. Minz, Antinuclear antibodies and their detection methods in diagnosis of connective tissue diseases: a journey revisited, *Diagnostic Pathology*. **4**(1), 1 (2009).
 - [32] *Quality assurance for the indirect immunofluorescence test for auto-antibodies to nuclear antigen (IF-ANA): Approved guideline*. vol. 16, NCCLS I/LA2-A. Wayne, PA (1996).
 - [33] P. Hobson, B. C. Lovell, G. Percannella, M. Vento, and A. Wiliem. Classifying hep-2 specimen images: A benchmarking platform. In *International Conference on Pattern Recognition (ICPR)* (2014).
 - [34] A. Wiliem, P. Hobson, R. F. Minchin, and B. C. Lovell, A bag of cells approach for antinuclear antibodies hep-2 image classification, *Cytometry Part A*. *In press.* (2015).
 - [35] G. Iannello, G. Percannella, P. Soda, and M. Vento, Mitotic cells recognition

- in hep-2 images, *Pattern Recognition Letters*. **(45)**, 136–144 (2014).
- [36] J. van Gemert, C. Veenman, A. Smeulders, and J. Geusebroek, Visual word ambiguity, *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*. **32**(7), 1271–1283 (2010).
 - [37] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid, Local features and kernels for classification of texture and object categories: A comprehensive study, *International Journal of Computer Vision*. **73**(2), 213–238 (2007).
 - [38] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2169–2178 (2006).
 - [39] D. G. Lowe, Distinctive image features from Scale-Invariant keypoints, *International Journal of Computer Vision*. **60**, 91–110 (2004).
 - [40] C. Sanderson and B. C. Lovell. Multi-region probabilistic histograms for robust and scalable identity inference. In *Lecture Notes in Computer Science (LNCS)*, vol. 5558, pp. 199–208 (2009).
 - [41] C. Liu, J. Yuen, and A. Torralba, SIFT flow: Dense correspondence across scenes and its applications, *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*. **33**(5), 978–994 (2011).
 - [42] C. Bishop, *Pattern Recognition and Machine Learning*. Springer (2006).
 - [43] Y. Wong, M. T. Harandi, C. Sanderson, and B. C. Lovell. On robust biometric identity verification via sparse encoding of faces: Holistic vs local approaches. In *IEEE International Joint Conference on Neural Networks*, pp. 1762–1769 (2012).
 - [44] A. Coates and A. Y. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *International Conference on Machine Learning (ICML)* (2011).
 - [45] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1794–1801 (2009).
 - [46] J. Tropp and S. Wright, Computational methods for sparse solution of linear inverse problems, *Proceedings of the IEEE*. **98**(6), 948–958 (2010).
 - [47] M. Aharon, M. Elad, and A. Bruckstein, K-SVD: an algorithm for designing overcomplete dictionaries for sparse representation, *IEEE Trans. Signal Processing*. **54**(11), 4311–4322 (2006).
 - [48] R. Rubinstein, A. M. Bruckstein, and M. Elad, Dictionaries for sparse representation modeling, *Proceedings of the IEEE*. **98**(6), 1045–1057 (2010).
 - [49] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2005).
 - [50] I. Kolár, J. Slovák, and P. Michor, *Natural operations in differential geometry*. Springer (1999). ISBN 3-540-56235-4.
 - [51] R. Bhatia, *Positive Definite Matrices*. Princeton University Press (2007).
 - [52] Y. M. Lui, Advances in matrix manifolds for computer vision, *Image and Vision Computing*. **30**(6-7) (2012).
 - [53] X. Pennec, Intrinsic statistics on Riemannian manifolds: Basic tools for

- geometric measurements, *Journal of Mathematical Imaging and Vision*. **25** (1), 127–154 (2006).
- [54] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache, Geometric means in a novel vector space structure on symmetric positive-definite matrices, *SIAM journal on matrix analysis and applications*. **29**(1), 328–347 (2007).
 - [55] S. Sra, Positive definite matrices and the s-divergence, *1110.1773v4* (2013). *Submitted*.
 - [56] A. C. Thompson, On certain contraction mappings in a partially ordered vector space, *Proceedings of the American Mathematical Society*. **14**, 438–443 (1963).
 - [57] A. Cherian, S. Sra, A. Banerjee, and N. Papanikolopoulos, Jensen-bregman logdet divergence with application to efficient similarity search for covariance matrices, *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*. **35**(9), 2161–2174 (2012).
 - [58] T. S. Lee, Image representation using 2d Gabor wavelets, *IEEE Trans. Pattern Analysis and Machine Intelligence (TPAMI)*. **18**, 959–971 (1996). ISSN 0162-8828.
 - [59] B. Kulic, K. Saenko, and T. Darrell, What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1785–1792 (2011).
 - [60] P. Hobson, B. C. Lovell, G. Percannella, M. Vento and A. Wiliem, Benchmarking human epithelial type 2 interphase cells classification methods on a very large dataset. *Artificial Intelligence in Medicine*, **65**(3), pp. 239–250 (2015).

Chapter 19

Representation Learning for Spoken Term Detection

P. Raghavendra Reddy¹, K. Sri Rama Murty¹ and B. Yegnanarayana²

¹*Department of Electrical Engineering
Indian Institute of Technology Hyderabad, India
ee12m1023@iith.ac.in, ksrm@iith.ac.in*

²*Indian Institute of Information Technology Hyderabad, India
yegna@iiit.ac.in*

Spoken Term Detection (STD), which refers to the task of searching for a user audio query in audio data is extremely significant for the management and monitoring of increasing volumes of audio data on the internet. It is affected by channel mismatch, speaker variability and differences in speaking mode/rate. Thus one of the main issues in STD is to devise a robust and speaker-invariant representation for the speech signal, so that the query and reference utterances can be matched in the new representation domain. In this chapter, the authors compare and contrast the supervised and unsupervised approaches to learn robust speech-specific representation for the STD task. Posterior representation of speech is used for developing the STD system with posterior features extracted in both supervised and unsupervised approaches.

19.1. Introduction

It is difficult to manage and monitor increasing volumes of data on the internet. Resources can be efficiently managed, if only the required information can be retrieved from this data. In the case of speech data, we need to search and locate spoken query words in large volumes of continuous speech. This task is termed as Spoken Term Detection (STD). Some of the applications of STD include speech data indexing [1], data mining [2], voice dialling and telephone monitoring.

Audio search can be broadly categorized into keyword spotting (KWS) and spoken-term detection (STD), depending on the domain (text or au-

dio) of the query supplied. In KWS system, the query word is supplied as text [3], [4]. Since the query word (text) and reference data (audio) are in two different domains, one of them needs to be transformed into the other domain to perform the search. Hence, the knowledge of the language and pronunciation dictionary is required to implement the KWS system. In the case of STD task, the query word is also supplied in the audio format [5], [6]. Since the reference and query are in the same domain, the STD task does not require the knowledge of pronunciation dictionary. However, it suffers from channel mismatch, speaker variability and differences in speaking mode/rate. One of the main issues in STD is to devise a robust and speaker-invariant representation for the speech signal, so that the query and reference utterances can be matched in the new representation domain. In this chapter, we compare and contrast the supervised and unsupervised approaches to learn robust speech-specific representation for the STD task.

In this work, we have used posterior representation of speech for developing the STD system. Posterior features are extracted in both supervised and unsupervised approaches. In supervised approach, a phoneme recognizer is trained using a combination of Hidden Markov Model (HMM) and Multilayer Perceptron (MLP) with labelled data, and the hybrid HMM-MLP model is employed to extract phonetic posterior features. In the absence of labelled data, the posterior features are extracted using two unsupervised methods, namely, Gaussian Mixture Model (GMM) and Gaussian Bernoulli Restricted Boltzmann Machine (GBRBm). Subsequence Dynamic Time Warping (DTW) is applied on the posterior features to search the query word in the reference utterance. The performance of the STD system is evaluated in terms of average precision, $P@N$, which indicates the number of correctly spotted instances of the query word out of total occurrences N of that word.

Rest of the chapter is organized as follows: Section 19.2 highlights the importance of feature representation for the STD task, and presents a survey of state-of-the-art approaches for arriving at a robust representation. Building STD system using phonetic posteriors, a representation learned using supervised approach, is discussed in Section 19.3. Unsupervised learning of representations from speech signals is discussed in Section 19.4 and Section 19.5 using Gaussian mixture models (GMM) and Gaussian Bernoulli Restricted Boltzmann Machine (GBRBm), respectively. Section 19.6 summarizes the important contributions of this study, and points to the important issues to be addressed in future.

19.2. Review of Approaches for STD

The task of STD can be accomplished in two important stages: (i) extracting a robust representation from the speech signal, and (ii) matching the representations obtained from reference and query utterances for detecting the possible locations.

19.2.1. Feature Representation for STD

Performance of STD system depends critically on the representation of the speech signal. The acoustic waveforms, obtained by measuring sound pressure levels, of a word uttered by two different speakers look completely different, and yet carry the same linguistic information. For example, the waveforms of the word “səmaɪk^hja” uttered by two different speakers is shown in Fig. 19.1(a). Though these two waveforms carry the same linguistic information, it is impossible to match them because of the natural variability (associated with fundamental frequency, rate of speech, context, etc.). Even though the similarity between them is better visible in the spectrogram representations, shown in Fig. 19.1(b), still it is difficult to match them using a machine.

The variability associated with the spectrograms, in Fig. 19.1(b), can be reduced by considering the average energies over a bands of frequencies. The mel-spaced filter bank energies, shown in Fig. 19.1(c), match better than the waveforms and their spectrograms. The cepstral coefficients derived from the mel-spaced filter bank energies, popularly called as mel-frequency cepstral coefficients (MFCCs), are the most widely used features in speech recognition systems [7]. Although they are well suited for the statistical pattern matching, like in HMMs for speech recognition, they may not be the best representation for template matching in the STD task. To illustrate this point, consider the task of searching the spoken query word “səmaɪk^hja” in the reference utterance “ra:stra:ni səmaɪk^hjəŋga unca:lən̩tu si:ma:n̩drəne:tələ.” The distance matrix ($\mathbf{D} = [d_{ij}]$) between the MFCC features of the reference and query utterances, for matched speaker condition, is shown in Fig. 19.3(a). The element d_{ij} in the distance matrix denotes the Euclidean distance between the MFCC features of i^{th} reference frame and j^{th} query frame. The distance matrices shown in Fig. 19.3(a) is a 3-dimensional representation, where x-axis denotes the sequence of reference frames, y-axis denotes the sequence of query frames, and intensity denotes the inverse of the pair-wise distances between reference and test frames.

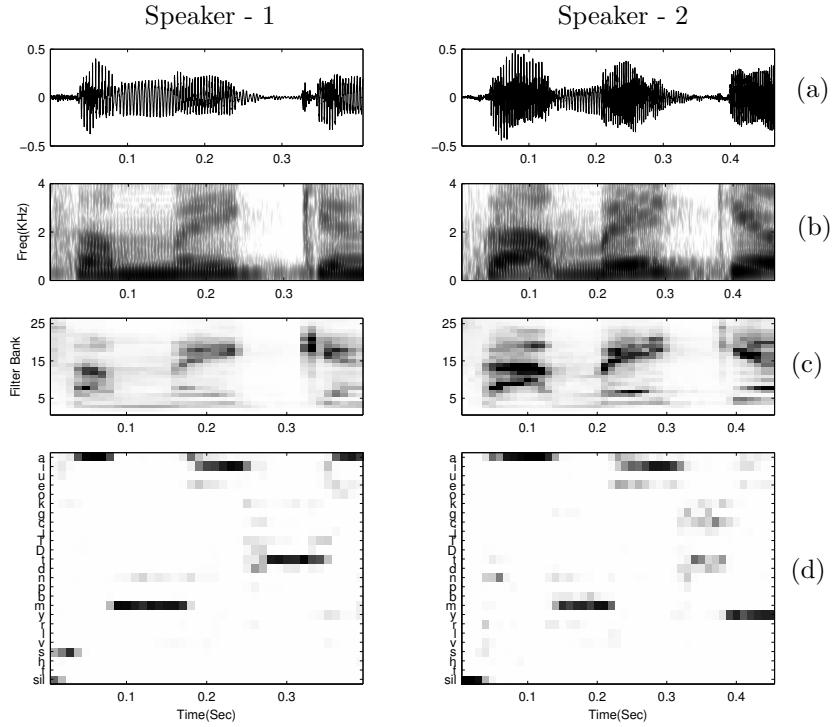


Fig. 19.1. Illustration of suitability of phonetic posteriors for template matching of the word “sõmaïk^hjõ” spoken by two different speakers. (a) Wave form (b) Spectrogram (c) Mel filter bank energies (d) MLP posteriors.

If the query word occurs in the reference utterance, then it is supposed to get reflected as an approximate diagonal path in the distance matrix. When the reference and query utterances are from the same speaker, there is a clear diagonal path in their distance matrix shown in Fig. 19.3(a). On the other hand, when the reference and query utterances are from different speakers, such a diagonal path can not be clearly spotted as in Fig. 19.3(b). This behaviour could be attributed to the speaker-specific nature of MFCC features, i.e., they do contain speaker-specific information. Notice that the MFCC features are used for speaker recognition also [8]. In the case of statistical pattern matching, the speaker-specific nature of MFCC features is normalized by pooling data from several different speakers. For STD also, we need to derive a robust speech-specific feature, from the speaker-independent component of MFCCs, for template matching.

A good feature representation for STD should be speech-specific, and at the same time, it should be robust to speaker and channel variability. Several attempts have been made to learn suitable representation from the data (MFCCs of any other feature) using statistical models and neural network models. Depending on the learning paradigm (supervised or unsupervised) and nature of the resultant representation (discrete or continuous), the representations for the STD are broadly classified into four categories. Table 19.1 presents the summary of different approaches to learn speech-specific representations for the STD task.

Table 19.1. Summary of representation learning approaches for STD.

	Discrete	Continuous
Supervised	<p>Sequence of phoneme-like labels</p> <p>Eg: LVCSR [9], Sub-word modelling [10]</p> <p>Pros: Good performance, faster matching</p> <p>Cons: Requires transcribed data, language knowledge and pronunciation dictionary</p>	<p>Sequence of posterior probability vectors of phonemes</p> <p>Eg: HMM [11], MLP [12], Hybrid HMM-MLP [5]</p> <p>Pros: Good performance, phoneme decoding is not required</p> <p>Cons: Requires transcribed data, slower matching</p>
Unsupervised	<p>Sequence of cluster indices obtained from acoustic similarity</p> <p>Eg: Vector Quantization (VQ) [13]</p> <p>Pros: Does not require transcribed data, simple to implement and faster matching</p> <p>Cons: Poor performance</p>	<p>Sequence of posterior vectors obtained from a trained statistical/neural network models</p> <p>Eg: GMM [14], ASM [15], GBRBM [16]</p> <p>Pros: Does not require transcribed data, optimal solution for low resource languages</p> <p>Cons: Intermediate performance, slower matching</p>

19.2.1.1. Learning discrete representations using supervised approaches

In this approach, a well trained large Vocabulary Continuous Speech Recognizer (LVCSR) [17], [9], [18] is used to convert the speech into text. Generally, it performs Viterbi search on word lattice to find the most probable sequence of words based on likelihoods of the trained acoustic models and the language model. It was observed that N-best hypothesis Viterbi search performs better than 1-best hypothesis, particularly when Word Error Rate (WER) of the system is high [19]. Variants of word lattices, namely, Word Confusion Network (WCN) [20], [21] and Position Specific Posterior Lattice

(PSPL) [22] were proposed to achieve good performance for IN-Vocabulary (INV) words. For spotting the locations of the query word in the reference utterance, text based searching methods were employed on the machine generated text transcripts [23]. This method is suitable mainly for high resource languages, as it requires large amount of transcribed speech data for training the language specific LVCSR system. Since these are word based recognizers, despite the high recognition accuracies for INV words, it suffers from Out-of-Vocabulary (OOV) problem. That is, it can not handle words which are not predefined in the pronunciation dictionary. In real-time implementation of query based search, encountering OOV words, like nouns, is very common. Hence, LVCSR based method can not be a practical solution even for high resource languages.

In order to address this issue, subword LVCSR based recognizers [10], [24], [17] were proposed, where recognition is done based on subwords instead of words. Since any word can be represented with basic subword (like phonemes) set, it is possible to recognize any word using subword recognizers. But the performance of subword-based recognizers deteriorates for INV in comparison to LVCSR based recognizers. The evidences from both phonetic and word lattices are combined to improve the performance [19], [25]. Generally, in sub-word based techniques, words are represented using phonetic lattices. In the matching stage, query search is accomplished by matching the phonetic lattices of the words.

19.2.1.2. Discrete representation using unsupervised approaches

Vector Quantization (VQ) can be employed to represent speech as a sequence of discrete symbols, in an unsupervised manner, i.e., without using the manual transcriptions [13]. In this method, clustering is performed on feature vectors extracted from speech signal and the set of mean vectors is stored as a codebook for representing speech. Feature vectors, derived from speech signal, can be represented as a sequence of codebook indices depending on their proximity to the cluster centres. In the matching stage, the sequence of codebook indices, obtained from the reference and query utterances, can be matched using approximate substring matching techniques [26]. Although this method does not require transcribed data and can be implemented with less complexity, its performance is significantly lower than the supervised approaches.

19.2.1.3. Continuous representation using supervised approaches

Conversion of speech signal into discrete sequence of phonemes, like in LVCSR, is less accurate because of the high degree of confusion among certain phonemes, especially among stop consonants. For example, the phoneme /k/ gets confused with /g/, and phoneme /d/ often gets confused with /t/ and /D/ and so on. This can lead poor transcription of reference and query utterances, resulting in lower STD performance. However, unlike speech recognition, STD task does not require the conversion of speech signal into sequence of phonemes. It was shown that phonetic posteriors, i.e., probability of the phoneme given the observed feature vector, are better suited for the STD task [27]. The posterior vector \mathbf{y} for an observed feature vector \mathbf{x} is defined as

$$\mathbf{y} = [P(c_1/\mathbf{x}) \ P(c_2/\mathbf{x}) \ P(c_3/\mathbf{x}) \ \cdots \ P(c_M/\mathbf{x})]^T \quad (19.1)$$

where $P(c_i/\mathbf{x})$ denotes probability of frame \mathbf{x} belonging to the i^{th} phoneme class c_i , and M denotes number of classes. The sequence of posterior vectors, commonly referred to as posteriorgram, forms a template representation of speech segment.

Phonetic posteriorgrams are extracted using well trained phoneme recognizers, built using hidden Markov models (HMMs) or Multilayer perceptrons (MLP) [28], [12]. It was shown that the phonetic posteriorgrams extracted using deep Boltzmann machines (DBM) are useful when limited amount of transcribed data is available [29]. In the next stage, dynamic time warping (DTW) is used to match the posteriorgrams extracted from the reference and query utterances.

Representing under resourced languages using well built multiple phone recognizers, each trained with one high resource language, was also explored by many researchers, and they were briefly summarized in [5], [6]. The features extracted from each recognizer for a given query, which can be from low resource language, were used to represent that query. However, all these approaches require labelled data in at least one language, which may not be feasible always.

19.2.1.4. Continuous representation using unsupervised approaches

Even though the supervised approaches are successful, they cannot be applied on a new language or under-resourced language, where manual transcriptions are not readily available. In such cases, unsupervised methods are preferred for posterior extraction [14], [15]. Most of the unsupervised

posterior extraction methods rely on estimating the joint probability density of the feature vectors of the speech signal. Gaussian mixture model (GMM) has been used for estimating the density function of the speech data and, there by, posterior extraction [14]. One drawback with this method is that GMM cannot model the temporal sequence in which the feature vectors have evolved. In order to address this issue, Acoustic Segment Models (ASM), which take temporal structure of speech into account, were proposed [15]. In this approach, each pseudo phoneme class is modelled as a HMM, where the class labels were obtained from pre-trained GMM. The classes represent pseudo phoneme like units, as they were obtained based on their acoustic similarity rather than their linguistic identity. It was shown that ASM outperforms well trained phoneme recognizer in language mismatched environment, and also GMM modelling. In the matching stage, a variant of DTW was used on GMM/ASM posteriors for searching the query word in the reference utterance. It was observed that the performance improves significantly by applying speaker normalization techniques, like Constrained Maximum Likelihood Linear Regression (CMLLR) and Vocal Tract Length Normalization (VTLN).

19.2.2. Template Matching of Feature Representations

Statistical behaviour of vocal tract system makes it impossible to generate two exactly similar speech segments in natural speech. So, no two speech utterances have equal durations of phonemes, and they are distributed non-linearly. In all the two stage approaches, matching stage plays crucial role in bringing out the excerpts of queries from continuous speech. Searching time and memory requirement are two main constraints in matching. Different matching methods are followed based on the representation of speech. In discrete representation, query words are represented as sequence of symbols or lattices. In [30], three methods for matching lattices were presented: Direct index matching, edit distance alignment and full forward probability. In the case of continuous representation, speech is represented as sequence of frames called as template. Matching templates was attempted in [31] using DTW. Degree of similarity of two given utterances can be approximated through cost of optimal path of DTW. Limitation of DTW is that durations of two utterances used for matching should be comparable, otherwise the computed similarity score denotes the closeness of long utterance with small utterance, like spoken word which are not comparable. Segmental DTW [32], subsequence DTW [33], unbounded DTW [34] and information

retrieval DTW (IR-DTW) [35] are few of the variants of DTW, which are tuned to search queries in continuous speech. Improvements to IR-DTW using hierarchical K-means clustering was proposed in [36].

In [32], segmental DTW was proposed for unsupervised pattern discovery. In this technique, starting point in one utterance was fixed, and DTW was applied with a constraint on the degree of warping path. Starting point was slid through the chosen utterance. In the next step, the obtained paths at each point were refined using length-constrained minimum average (LCMA) algorithm [37] to get minimum distortion segments. In [14], modified segmental DTW, where LCMA is not applied on the paths, was used to accomplish STD goal. As this method requires DTW at periodic intervals of time in either of the two utterances, it requires high computation and memory resources, making it practically impossible on large archives.

In [33], subsequence DTW was proposed for STD task where calculation of accumulated matrix is different from conventional DTW. In the conventional DTW, dissimilarity values along the reference utterance were accumulated forcing the backtracked path to reach first frame. Using the fact that query can start from any point in the reference utterance, accumulation of values along first row of the reference utterance was not done, which makes the back-traced path to end at any point in reference utterance. Path was backtracked from minimum accumulated distortion.

In [35], IR-DTW was proposed for the STD task, in which memory requirement was reduced by avoiding calculation of full distance matrix. Starting indices in both query and reference utterances were chosen, based on exhaustive similarity computation. By using non-linear subsequence matching algorithm, ending indices were found, and the best matching segments were filtered. This algorithm can be scaled up for large amounts of data with reduced memory requirements. When query term contains multiple words, or if query term is not present exactly in the reference utterance, but parts of query term are jumbled in the reference utterance (like in QUESST task in MediaEval 2015), then this method is useful. This method was further improved using K-means clustering algorithm instead of exhaustive distance computation [36]. By imposing constraints on similarity score on matching paths, searching can be made faster as in [38], [39], [40].

19.3. Posterior Extraction using Hybrid HMM-MLP

A combination of generative and discriminative models have proven to be effective for complex classification tasks, like speech recognition [41]. Generative models estimate joint density of the input data, while discriminative models capture the boundaries between the classes. Examples for generative models include GMM, HMM, Restricted Boltzmann Machines (RBM) and Gaussian- Bernoulli RBM (GBRB). Examples for discriminative models include Support Vector Machines (SVM), and Multi Layer Perceptron (MLP). It was shown that a combination of HMM-MLP hybrid modelling is better suited for phoneme recognition, than either one of them alone [42]. In the HMM-MLP hybrid modelling, HMM is aimed at handling the varying length patterns, while the MLP is aimed at estimating the non-linear discriminant functions between the phoneme classes. In this approach, the state emission probabilities of the HMM are replaced with the posterior probabilities of the states estimated by the MLP. The phonetic posteriograms, estimated from HMM-MLP hybrid modelling, can be used as a representation for STD task.

19.3.1. *Speech Segmentation using HMM*

HMMs are doubly stochastic models, and can be used to model non-stationary signals like speech. Assuming that a state represents a specific articulatory shape of the vocal tract, the speech signal (observation sequence) can be represented by a sequence of states. Here, the state sequence is not known (hidden) to us. Through HMM training, parameters of each state are obtained to capture the statistical properties of speech signal in that state. Generally, each phoneme is modelled using a three-state left-right HMM, assuming that the phoneme is produced in 3 phases. For example, the production of a stop-consonant consist of three main phases, namely, closure phase, burst phase and transition phase into succeeding phoneme. More number of states can also be used for better modelling, but it requires large amount of data for training. In this evaluation, each phoneme is modelled as a three-state HMM with 64 Gaussian mixtures per state. Labelled speech data is used to estimate the parameters of the HMM, which include the initial probabilities, emission probabilities and state-transition matrices, using Baum-Welch algorithm [43]. The trained HMM models are used to align the manual transcriptions with the speech data, to obtain the phoneme boundaries. The phoneme boundaries ob-

tained from forced alignment are used for training a Multi Layer Perceptron (MLP).

19.3.2. Extraction of Phonetic Posteriors using MLP

A multilayer perceptron (MLP) is a feed forward artificial neural network model that maps sets of inputs onto a set of appropriate outputs. An MLP consists of multiple layers of nodes in a directed graph, as in Fig. 19.2, with each layer fully connected to the next one. Each node, in the hidden and output layers, is a neuron (or processing element) with a non-linear activation function. Sigmoid and hyperbolic tangent activation functions are typically used in the hidden layers, while softmax activation function is used in the output layer. The output of the softmax function can be interpreted as posterior probability of the class given the input frame. The weights of the network can be learnt, using back-propagation algorithm, by maximizing the cross entropy between the estimated posteriors and actual phoneme labels [44].

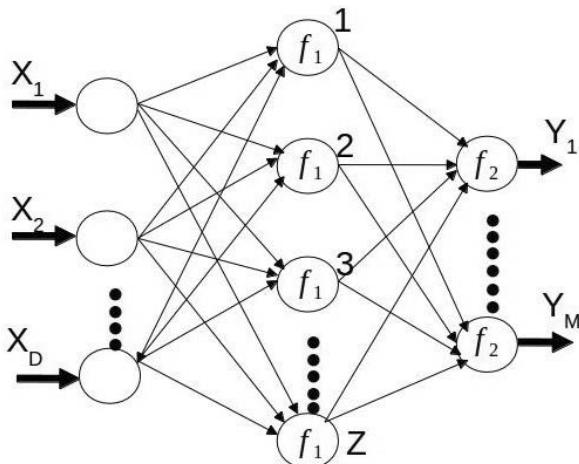


Fig. 19.2. MLP Network. X and Y are the input and output vectors respectively. D , Z and M denote the number nodes at the input, hidden and output layer respectively. f_1 and f_2 denotes the activation functions at hidden and output layer respectively.

Unlike GMM, an MLP can be trained with higher dimensional correlated data. Hence, context information can be learnt using MLP by presenting concatenated speech frames as input. A context of 13-frames is

used with 39-dimensional MFCC features to form a 507 (39×13) dimensional input feature vector. Phoneme labels obtained from the HMM forced alignment are used as output classes. An MLP with single hidden layer, having 1000 sigmoid units, is trained to map the input feature vectors to the phoneme label. The performance of HMM-MLP hybrid approach was evaluated on 5 hours of Telugu broadcast news data, in which 3 hours of data was used for training and the remaining 2 hours was used for testing the system. The performance of HMM-MLP hybrid approach is shown in Table 19.2 for different configurations of phoneme groupings as given in Table 19.3. The performance of HMM system alone is also given for comparison. The HMM-MLP hybrid system is consistently better than the HMM alone. As the number of classes increase, there is a gradual drop in the recognition accuracy.

Table 19.2. Recognition accuracy for different number (M) of phoneme classes.

M	HMM	HMM-MLP
6	77.88	81.87
15	70.6	76.02
25	69.51	74.24
45	62.68	69.11

Table 19.3. Grouping of the phonemes into different classes.

45 classes	a	æ	i	ɛ	j	u	ɯ	e	ɛ	o	ɑ	v	f	ʃ	s	h	t	m	n	k	˥	g	ɣ	c	ʎ	z	˥	ʃ̥	˥	ʒ	˥	t̥	˥	d	˥	d̥	˥	p	˥	b	˥	r	˥	l	˥	sil
25 classes	a		i		u		e		o		v		s		h		f		m		n		k		g		c		z		ʃ		t		d		p		b		r		l		sil	
15 classes	a		i		u		e		o		V[Vowel]		P[Consonants]		N[Nasal]		G[Glotal]		P[Palatal]		R[Retroflex]		D[Dental]		B[Bilabial]		T[Tongue tip]		L[Lip]		S[Sibilant]		T[Trill and Liquid]		sil[Silence]											
6 classes																																														

19.3.3. STD using Phonetic Posterior

The MFCC features extracted from every 25 ms frame of speech signal is converted into phonetic posterior representation using the trained MLP. Since phonetic posteriors are obtained from the MLP, trained with large amount of speech data collected from several speakers, they are more robust to speaker variability. Hence they are better suited for the STD, than the raw MFCC features. Phonetic posteriors of word “səmaɪkʰjø” spoken by two different speakers are shown in Fig. 19.1 which can be better matched. Speaker invariant nature of phonetic posteriors is illustrated, in Figs. 19.3(c) and 19.3(d) using the task of searching for a query word “səmaɪkʰjø” in the reference utterance “ra:stra:ní səmaɪkʰjøŋga unca:lən̩tu

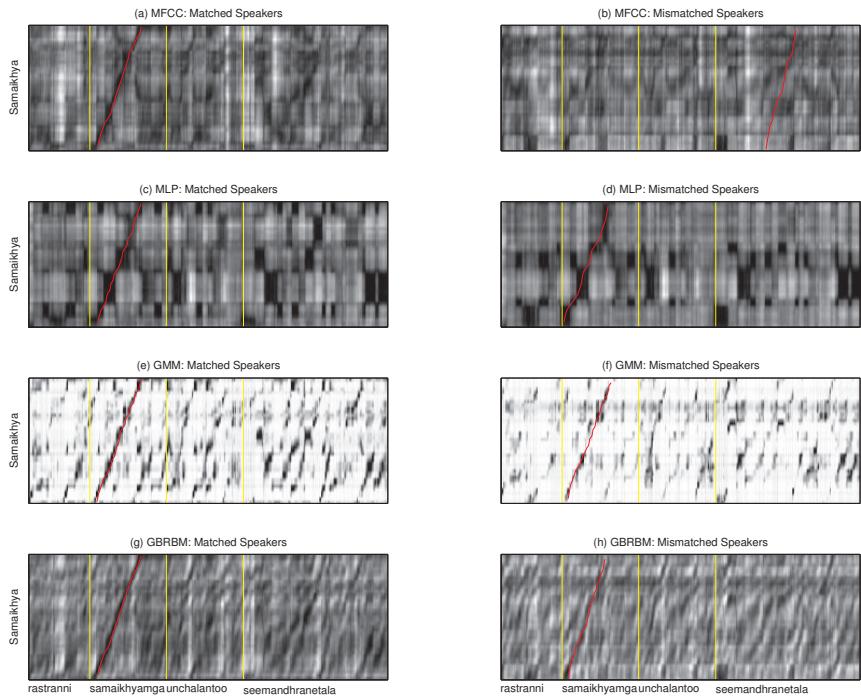


Fig. 19.3. Illustration of effectiveness of posterior features for STD over MFCC. Distance matrix along with DTW path computed from (a) MFCC features for matched speakers, (b) MFCC features for mismatched speakers, (c) MLP posteriors for matched speakers, (d) MLP posteriors for mismatched speakers, (e) GMM posteriors for matched speakers, (f) GMM posteriors for mismatched speakers, (g) GBRBM posteriors for matched speakers, (h) GBRBM posteriors for mismatched speakers.

si:ma:n̥drøne:tølø.” We have considered two cases: the query word is from the same speaker as the reference utterance and the query word is from a different speaker. Euclidean distance and symmetric Kullback-Leibler Divergence (KL divergence) are used to find the distance between two MFCC and two posterior features, respectively. The distance matrices computed from MFCC features are shown in Figs. 19.3(a) and 19.3(b); and the distance matrices computed from posterior features are shown in Figs. 19.3(c) and 19.3(d). In the case of matched speakers, there is a DTW path at correct location, indicating the presence of query word in the reference utterance, in distance matrices computed from both MFCC and posterior features. When the speakers do not match, the desired location of query

word is successfully found in the distance matrix computed from the MLP posterior features, but not from the MFCC features. This case study depicts the speaker-invariant nature of posterior features, and thereby their effectiveness in STD.

Subsequence DTW is employed to match the posterior features extracted from the reference and query utterances, and detect the possible matching locations of query in the reference utterance. Each vector in posteriorgram can be interpreted as a probability distribution. If any element in posterior vector \mathbf{y} is zero then the KL divergence with any other vector is infinity, which is not desired. To avoid this, smoothing method is suggested in [12]. So, the new posterior vector \mathbf{y}_{new} is

$$\mathbf{y}_{new} = (1 - \lambda)\mathbf{y} + \lambda\mathbf{u}$$

where \mathbf{u} is a sample vector drawn from uniform distribution \mathcal{U} , and λ is smoothing constant. For all experiments in this study, λ is empirically chosen as 0.01. Performance of STD system is evaluated in terms of average precision ($P@N$), where N is number of occurrences of the query in the reference utterance. The evaluation metric $P@N$ is calculated as proportion of query words located correctly in top N hits from reference utterance. Detected location is chosen as hit, if it overlaps more than 50% with reference location.

This method is evaluated on 1 hour of Telugu broadcast news data with 30 query words spliced from continuous speech data. The performance of the STD system obtained with different phoneme classes is given in Table 19.4. Even though the phoneme recognition accuracy increased with reducing the number of phoneme classes, it did not result in improved STD. There is a significant decrease in $P@N$ from 15 to 6 classes. Since several phonemes are grouped together into a single class, the number of false matches increases resulting in poor performance. The best performance was achieved with 25 phoneme classes, in which the aspirated and unaspirated stop constants produced at the same place of articulation are grouped together. Hence 25 phoneme classes are used for all further studies in this chapter.

Table 19.4. Average performance of STD obtained with different phoneme classes.

Metric	6 classes	15 classes	25 classes	45 classes	raw MFCCs
$P@N$	44.05	77.65	80.13	72.36	45.68
$P@2N$	55.68	86.50	89.13	80.57	54.91
$P@3N$	59.17	88.10	90.75	82.39	60.81

19.3.4. Effect of Speaking Mode

The experiments, reported in the previous section, were conducted on 30 queries spliced from continuous read speech. In this section, the performance of the STD system is evaluated on the query words recorded from 20 native Telugu speakers in an isolated manner. It is observed that the duration of the query words recorded in isolated manner is almost double the duration of those spliced from continuous speech [45]. Since the query words are recorded in a different environment, there is channel mismatch between the reference and query words. Both these factors (duration and channel mismatch) lead to significant drop in the STD performance. In order to mitigate the effect of duration mismatch, experiments are conducted using different constraints on the warping path, as shown in Fig. 19.4. The weights (a, b, c, d, e) can be used to modify the shape of the warping path. A vertical path can be favoured by decreasing d and e , where as a horizontal path can be favoured by decreasing a and b . A diagonal path can be favoured by decreasing c . In the case of isolated queries, whose durations are much longer, a vertical path should be favoured. The best performance with isolated queries was obtained with weights (2, 3, 2, 1, 1). In order to normalize the channel effects, cepstral mean subtraction and variance normalization are performed for every utterance. The average performance of STD, for both spliced and isolated queries, is given in Table 19.5. The performance of the STD system with isolated queries is almost 25% less than that of with the spliced queries.

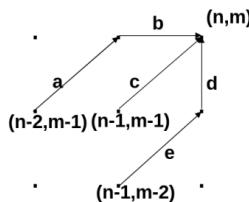


Fig. 19.4. Subsequence DTW path with local weights (a, b, c, d, e).

Table 19.5. Comparison of STD performance, with queries spliced from continuous speech and queries recorded in isolation.

Metric	$P@N$	$P@2N$	$P@3N$
Spliced from read data	80.49	88.61	90.10
Isolated recordings	56.02	66.70	69.66

19.4. Posterior Extraction using GMM

Even though the performance of supervised methods is satisfactory, they require labelled data to train the models, which may not be available always, particularly for low resource languages. In this scenario, unsupervised approaches can be a promising solution. In this study, two approaches are explored for posterior feature extraction in the absence of labelled data. Generative models, namely GMM and GBRBM, are employed for unsupervised posterior feature extraction.

Mixture models capture the underlying statistical properties of data. In particular, GMM models the probability distribution of the data as a linear weighted combination of Gaussian densities. That is, given a data set $X = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, the probability of data X drawn from GMM is

$$p(X) = \sum_{i=1}^M w_i \mathcal{N}(X/\mu_i, \Sigma_i) \quad (19.2)$$

where $\mathcal{N}(\cdot)$ is Gaussian distribution, M is number of mixtures, w_i is the weight of the i^{th} Gaussian component, μ_i is its mean vector and Σ_i is its covariance matrix. The parameters of the GMM $\theta_i = \{w_i, \mu_i, \Sigma_i\}$ for $i = 1, 2, \dots, M$, can be estimated using Expectation Maximization (EM) algorithm [46].

Figure 19.5 illustrates the joint density capturing capabilities of GMM, using 2-dimensional data uniformly disturbed along a circular ring. The red ellipses, superimposed on the data (blue) points, correspond to the locations and shapes of the estimated Gaussian mixtures. In the case of 4-mixture GMM, with diagonal covariance matrices, the density was poorly estimated at odd multiples of 45° , as shown in Fig. 19.5(a). As the number of mixtures increases, the density is better captured as shown in Fig. 19.5(b). Since the diagonal matrices cannot capture correlations between dimensions, the curvature of the circular ring is not captured well. In the case of diagonal covariance matrices, the ellipses are aligned with the xy-axes as shown in Fig. 19.5(a) and Fig. 19.5(b). The density estimation can be improved using full covariance matrices, as shown in Fig. 19.5(c). However, this improvement comes at the expense of increased number of parameters and

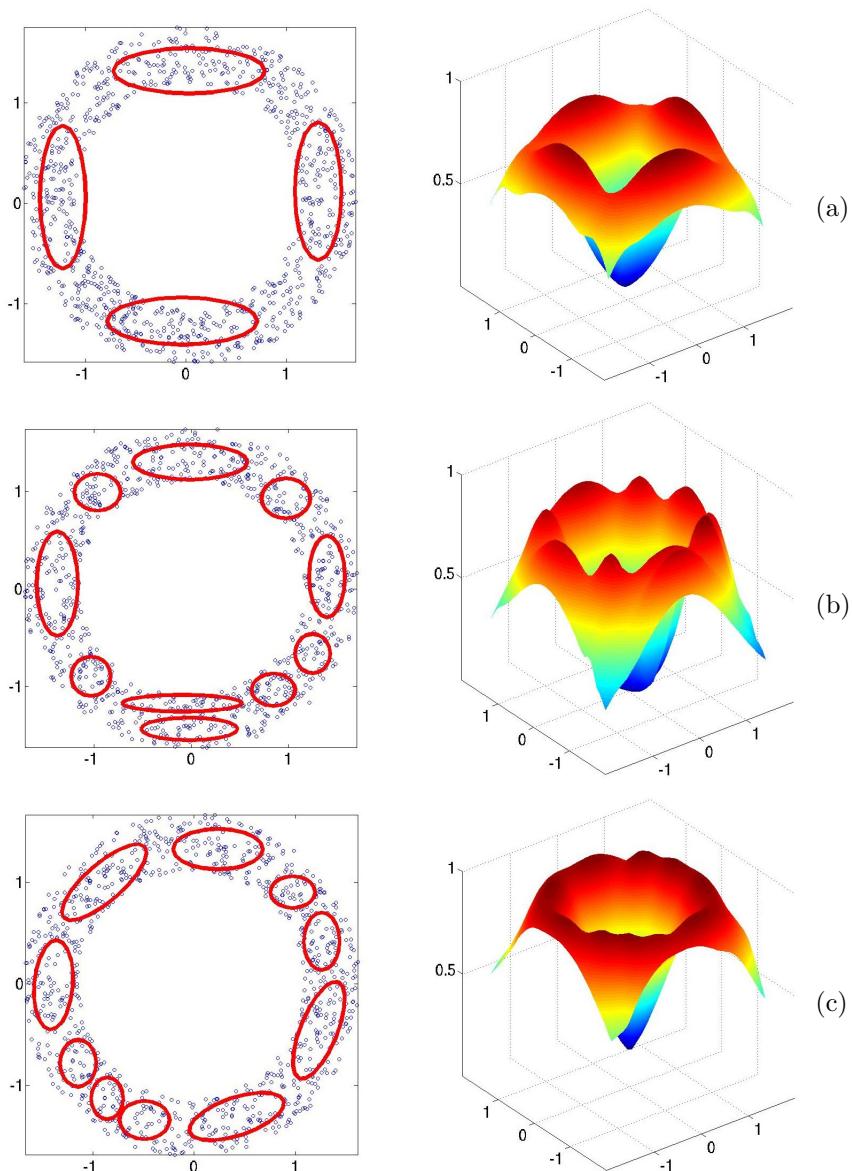


Fig. 19.5. Illustration of distribution capturing capability of GMM. GMM trained with diagonal covariance matrices (a) 4-mixtures (b) 10-mixtures and (c) 10-mixture GMM trained with full covariance matrices.

computation. We need to estimate $M(2D+1)$ parameters for an M-mixture GMM, with diagonal covariances, where D is the dimension of the data. For a GMM with full covariance matrices, we need to estimate $M(0.5D^2 + 1.5D + 1)$ parameters, which in turn requires large amount of data.

Given a trained GMM and a data point \mathbf{x} , the posterior probability that it is generated by the i^{th} Gaussian component c_i can be computed using the Bayes' rule as follows:

$$P(c_i/\mathbf{x}) = \frac{w_i \mathcal{N}(\mathbf{x}/\mu_i, \Sigma_i)}{p(\mathbf{x})} \quad (19.3)$$

The vector of posterior probabilities for $i = 1, 2, \dots, M$ is called Gaussian posterior vector. Gaussian posterior representation was found be better suited for STD than the MFCC coefficients [14], [47].

19.4.1. STD using Gaussian Posteriors

In this study, a GMM is built by pooling MFCC feature vectors extracted from 5 hours of speech data. Using this model, the reference and query utterances are represented as a sequence of GMM posterior features. The distance matrices computed from GMM posteriors, for matched and mismatched speaker conditions, are shown in Fig. 19.3(e) and Fig. 19.3(f), respectively. It can be observed that the GMM posteriograms are better at handling the speaker variability.

Subsequence DTW is used to match the GMM posteriograms of reference and query utterances, to perform the STD task. The performance of the STD system, with varying number of Gaussian components, is given in Table 19.6 with symmetric KL divergence as distance measure. The performance of the system is improved as the number of mixtures increase. It can be seen that for GMM trained with less number of mixtures the STD performance is low, which could be due to clustering of multiple phonemes under same mixture. The lower performance of 128-mixture GMM may be attributed to association of each phoneme class with more than one mixture. It can be clearly seen that GMM posteriors are more robust across speakers, giving better STD performance over MFCC.

Table 19.6. Effect of number of mixtures in GMM on STD performance.

Metric	MFCC	GMM-16	GMM-32	GMM-64	GMM-128
$P@N$	45.68%	46.2%	48.9%	53.1%	52.8%

19.5. Posterior Extraction using GBRBM

A Restricted Boltzmann machine (RBM) is an undirected bipartite graphical model with visible and hidden layers. In contrast to a Boltzmann machine, intra-layer connections do not exist in RBM, and hence the word *restricted*. In an RBM, the output of a visible unit is conditionally Bernoulli given the state of hidden units. Hence the RBM can model only binary valued data. On the other hand in a GBRBM, the output of a visible unit is conditionally Gaussian given the state of hidden units, and hence it can model real valued data. Both in RBM and GBRBM, the output of a hidden unit is conditionally Bernoulli, given the state of visible units, and hence can assume only binary hidden states. Since the same binary hidden state is used to sample all the dimensions of the visible layer, GBRBM are capable of modelling correlated data.

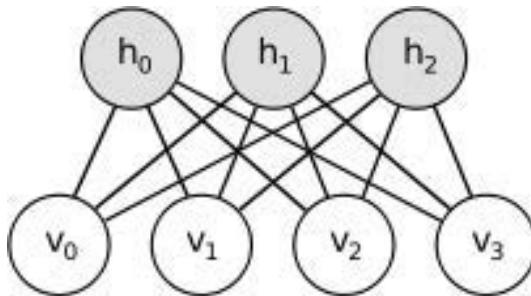


Fig. 19.6. Network architecture of a Restricted Boltzmann Machine.

A GBRBM can be completely characterized by its parameters, i.e., weights, hidden biases, visual biases and variances of the visible units. The GBRBM associates an energy for every configuration of visible and hidden states. The parameters of the GBRBM are estimated such that the overall energy of GBRBM, over the ensemble of training data, reaches a minima on the energy landscape. The energy function for GBRBM, for a particular configuration of real-valued visible state vector \mathbf{v} and binary hidden state vector \mathbf{h} , is defined as [48]

$$E(\mathbf{v}, \mathbf{h}) = \sum_{i=1}^V \frac{(v_i - b_i^v)^2}{2\sigma_i^2} - \sum_{j=1}^H b_j^h h_j - \sum_{i=1}^V \sum_{j=1}^H \frac{v_i}{\sigma_i} h_j w_{ij}, \quad (19.4)$$

where V and H are total number of visible and hidden units, v_i is the state of i^{th} visible unit, h_j is the state of j^{th} hidden unit, w_{ij} is the weight connecting the i^{th} visible unit to the j^{th} hidden unit, b_i^v is the bias on the

i^{th} visible unit, b_j^h is the bias on the j^{th} hidden unit, σ_i is the variance of the i^{th} visible unit.

The joint density of the visible and hidden unit states is related to the energy of the network as

$$p(\mathbf{v}, \mathbf{h}) \propto e^{-E(\mathbf{v}, \mathbf{h})} \quad (19.5)$$

The parameters of the GBRBM are estimated by maximizing the likelihood of the data. The updates for the parameters can be estimated using Contrastive Divergence (CD) algorithm [49], as follows:

$$\begin{aligned}\Delta w_{ij} &\propto \left\langle \frac{v_i h_j}{\sigma_i} \right\rangle_{data} - \left\langle \frac{v_i h_j}{\sigma_i} \right\rangle_{recall} \\ \Delta b_i^v &\propto \left\langle \frac{v_i}{\sigma_i^2} \right\rangle_{data} - \left\langle \frac{v_i}{\sigma_i^2} \right\rangle_{recall} \\ \Delta b_j^h &\propto \langle h_j \rangle_{data} - \langle h_j \rangle_{recall} \\ \Delta \sigma_i &\propto \langle \gamma \rangle_{data} - \langle \gamma \rangle_{recall}\end{aligned}$$

where

$$\gamma = \frac{(v_i - b_i^v)^2}{\sigma_i^3} - \sum_{j=1}^H \frac{h_j w_{ij} v_i}{\sigma_i^2}$$

and $\langle \cdot \rangle_{data}$ denotes expectation over the input data, and $\langle \cdot \rangle_{recall}$ denotes expectation over its reconstruction.

During each cycle of CD, the energy associated with the joint configuration of visible and hidden states is supposed to decrease, although there is no theoretical guarantee. After a large number of iterations, the expectation of the energy does not change any more, indicating thermal equilibrium of the network. At thermal equilibrium, the GBRBM models the joint density of the training data. The trained GBRBM model is capable of generating the data points which resemble the training data.

The distribution capturing capability of GBRBM is illustrated, in Fig. 19.7, with 2-dimensional data uniformly distributed along a circular ring. First column shows the mean of the unbiased samples generated by the model, second column shows the reconstructed data points, and third column shows the estimated density function. Input data points are plotted as blue ‘o’ and reconstructions are plotted as red ‘+’. A GBRBM with different number of hidden units is trained, to capture the joint density of this data, for 200 cycles using CD. The number of hidden units plays an important role in capturing the distribution of input data. For a GBRBM with H hidden units, the hidden state vector can take at most 2^H binary

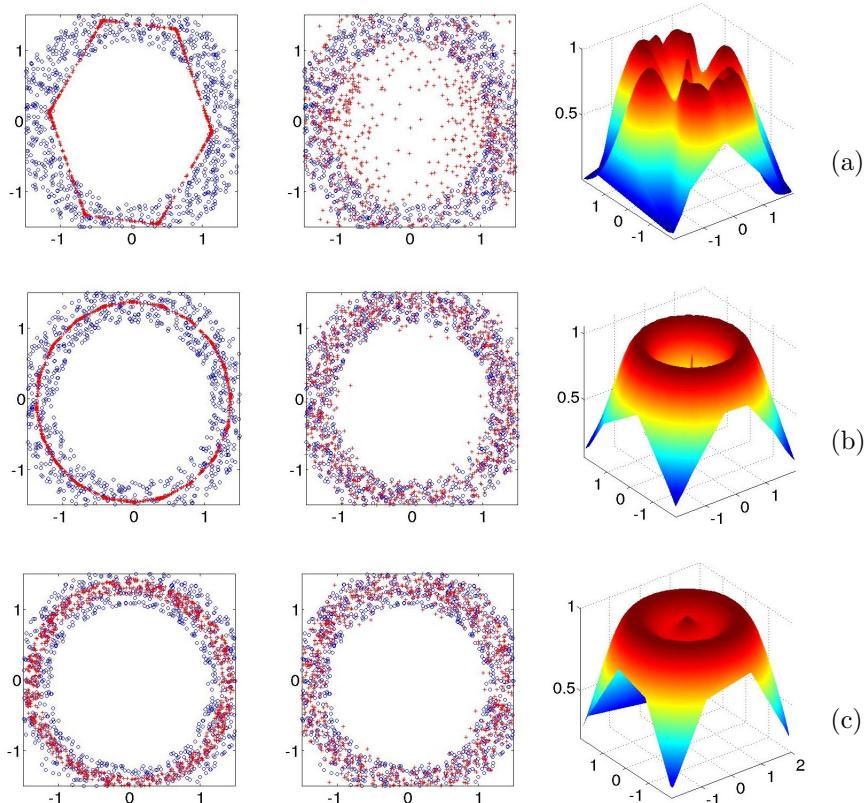


Fig. 19.7. Illustration of distribution capturing capability of GBRBM. Left: Original training data (blue), and mean of the unbiased samples generated by trained GBRBM, Middle: Original training data (blue), and unbiased samples generated by GBRBM, Right: captured density plot. GBRBM plots trained with (a) 3 (b) 10 (c) 200 hidden layer neurons.

configurations. However, only a few of those 2^H hidden states sustain at the thermal equilibrium of the network. Hence, the reconstructed visible state can take a maximum of 2^H different mean vectors. The mean visible layer activations, for a GBRBM with 3 hidden units is shown in Fig. 19.7(a). In this case the mean of the circular ring is approximated by a hexagon, i.e., with 6 (out of $< 2^3$ possible) stable states at thermal equilibrium. As the number of hidden units increases, the number of possible stable states also increases, leading to a better approximation of the mean of the input data.

The mean of the unbiased samples generated by a GBRBM with 10 hidden units, in Fig. 19.7(b), faithfully estimated the mean of the circular ring. When the number of hidden units is further increased to 200, the mean activations are spread over input data leading to an overfit. The data distributions captured by GBRBMs, with different hidden units, are shown in the third column of Fig. 19.7. It is clear that the GBRBM with 10 hidden units has captured the input distribution better.

The variance parameters of the visible units also influence the distribution capturing capabilities of the GBRBM. Usually, when the GBRBM is used to pre-train the first layer of an MLP the variances are simply set to one. However, variance learning is necessary when GBRBM is used to capture the joint density of the input data. Figure 19.8 shows the density captured by GBRBM, with 10 hidden units, trained without variance parameter. Clearly, without variance learning, the GBRBM failed to capture the underlying joint density of the data.

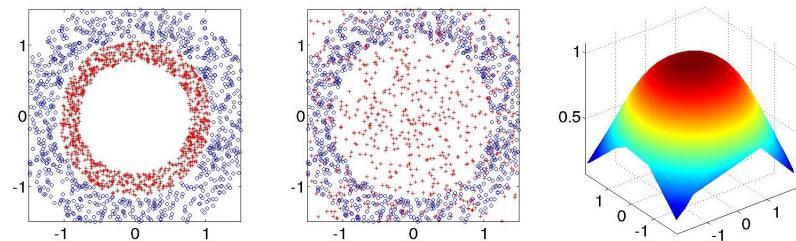


Fig. 19.8. Importance of variance learning: Left: Original training data (blue), and mean of the unbiased samples generated by trained GBRBM, Middle: Original training data (blue), and unbiased samples generated by GBRBM, Right: captured density plot. Plots are generated from GBRBM trained with 10 hidden layer neurons.

19.5.1. STD using GBRBM Posteriors

A GBRBM, with 50 hidden units is trained, using 39-dimensional MFCC features to capture the density of the speech data in the acoustic space. The marginal densities of the original (blue) and estimated (red) MFCC features are shown in Fig. 19.9. The marginal densities of the first 12 MFCC features is shown to illustrate the effectiveness of GBRBM in capturing the joint density of the data. In this chapter, the state of the hidden units, at thermal equilibrium, is used as a feature for STD task. The probability that the j^{th} hidden neuron assumes a state of ‘1’, given the state of the

visible units (MFCC features) is computed as

$$P(h_j = 1 \mid \mathbf{v}) = \text{sigmoid} \left(\sum_{i=1}^V \frac{v_i}{\sigma_i} w_{ij} + b_j^h \right) \quad (19.6)$$

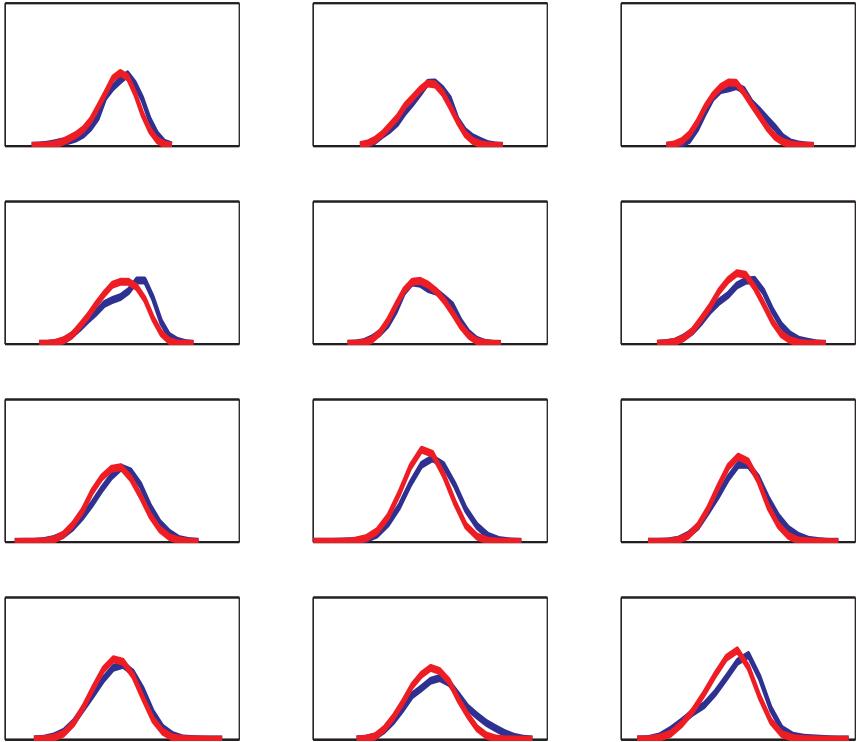


Fig. 19.9. Comparison of original (red) and estimated (blue) marginal densities of first 12 dimensions of MFCC features.

The posterior probability vector, representing the state of all the hidden units, is used to match the reference and query utterances. The distance matrices computed from GBRBM posteriors, with symmetric KL divergence, for matched and mismatched speaker conditions are shown in Fig. 19.3(g) and Fig. 19.3(h), respectively. In the case of mismatched speakers, the distance matrix computed from GBRBM posteriors helped to spot correct location of the query word as shown in Fig. 19.3(h), which is not the case using distance matrix computed from MFCC features as shown

in Fig. 19.3(b). Hence the GBRBM posterior representation is better than MFCC features for STD task.

The performance of STD system built with GBRBM posteriors is given in Table 19.8. GBRBM-NV denotes GBRBM trained without variance parameter learning. In this case, STD performance is no better than MFCC which denotes that density of input data is not captured. The performance of the GBRBM posteriors is slightly better than the GMM posteriors which can be attributed to ability of GBRBM to exploit the dependencies between dimensions of input data. Since GMM and GBRBM are two different unsupervised data modeling techniques, the evidences from both these systems are combined linearly. The performance of the combined system is better than the performance of either of the systems alone. However, the performance of the combined system is much lower than the performance of the phonetic posteriors obtained from HMM-MLP hybrid model. This is because the phonetic posteriors are obtained using a supervised approach, while the GMM and GBRBM posteriors are unsupervised approaches.

Table 19.7. Results of STD with various number of hidden neurons.

Metric	GBRBM hidden neurons			
	30	50	70	90
<i>P@N</i>	50.83%	57.64%	56.58%	56.43%

Table 19.8. Performance comparison (%) of STD systems built using different posterior representations.

Metric	MFCC	GMM	GBRBM-NV	GBRBM	GBRBM+GMM	HMM-MLP
<i>P@N</i>	45.68%	53.10%	45.84%	57.64%	59.91%	80.49%

The performance of the STD system, built using different posterior features, on 30 query words from Telugu language is presented in Table 19.9. Assuming that the probability of misclassification is same for all the syllables, miss rate of longer query words is less compared to smaller query words. On an average, this can be observed in the Table 19.9 for all representations. For longer query words, the performance is almost similar, with all the three representations, but for smaller query words HMM-MLP posterior features perform better than GMM and GBRBM posteriors.

Table 19.9. Query words (written in IPA) with their $P@N(\%)$ for Telugu language.

Query word	HMM-MLP	GMM	GBRBM	Query word	HMM-MLP	GMM	GBRBM
prənəbmuk ^h ərji	99.75	100	100	digvijeojsing	82.78	50	60
telənga:nə	83.50	90	90	adjoks ^h ura:lu	76.96	50	50
səma:ue:səm	88.97	84.09	86.36	prəb ^h utuəm	71.43	48.57	57.14
sailəja:na:t ^h	84.64	80	60	ad ^h ika:rulu	85.31	42.85	64.29
alpəpi:dənəm	84.62	76.92	69.23	haidra:ba:d	83.57	42.85	71.43
pa:rələment	88.24	76.47	76.47	kəm:ṭi	58.82	35.29	35.29
bənga:la:k ^h a:təm	81.75	75	75	en <small>◦</small> nikəlu	72.25	35.13	40.54
kangres	78.00	74	78	erpa:tu	63.38	31.8	40.91
ra:j:i:na:ma	85.19	70.37	59.26	va:ta:vərənəm	67.00	30	50
ne:pət ^h jəm	62.69	69.23	76.92	vib ^h əjənə	71.43	28.57	33.33
pənca:jəti	76.62	68.96	82.76	səmaik ^h jə	93.55	22.58	54.84
so:mija:ga:nd ^h i	90.83	66.66	83.33	dilli:	50.00	20.83	41.67
po:liŋg	63.75	62.5	75	viuəra:lu	80.00	20	59.91
kirənkuma:rredđi	95.53	57.14	85.71	rupa:ji	70.00	20	40
nirnəjəm	83.33	55.55	63.89	məntri	32.73	14.28	12.70

19.6. Summary and Conclusions

In this study, we have presented the development of a spoken term detection system for Telugu Language. Subsequence DTW is employed to search for a query word in the reference utterance. The representation of reference and query utterances plays a crucial role during the search. In this chapter, three different representation techniques are investigated, namely phonetic posteriors, GMM posteriors and GBRBM posteriors. The phonetic posteriors, obtained from HMM-MLP phoneme recognizer, requires large amount of manually labelled data. On the other hand, the GMM posteriors and the GBRBM posteriors can be obtained from unlabelled speech data. It is observed that the performance of phonetic posteriors is much better than the performance of the GMM and GBRBM posteriors. However, its application is limited since it requires labelled data. Future efforts will focus on improving the unsupervised feature representation techniques, using sequence and context information.

References

- [1] J. Foote, An overview of audio information retrieval, *Multimedia Syst.* **7**(1), 2–10 (Jan., 1999). ISSN 0942-4962. doi: 10.1007/s005300050106.
- [2] A. J. K. Thambiratnam. *Acoustic keyword spotting in speech with applications to data mining*. PhD thesis, Queensland University of Technology (2005).
- [3] J. G. Fiscus, J. Ajot, J. S. Garofolo, and G. Doddington. Results of the 2006 spoken term detection evaluation. In *Proc. SIGIR Special Interest Group on Information Retrieval Workshop, Amsterdam, Netherlands*, vol. 7, pp. 51–57 (2007).
- [4] J. Tejedor, D. T. Toledano, X. Anguera, A. Varona, L. F. Hurtado, A. Miguel, and J. Cols, Query-by-example spoken term detection albayzin 2012 evaluation: overview, systems, results, and discussion, *EURASIP Journal on Audio, Speech and Music Processing*. **23**, 1–17 (September, 2013).
- [5] F. Metze, X. Anguera, E. Barnard, M. Davel, and G. Gravier. The spoken web search task at mediaeval 2012. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, British Columbia, Canada*, pp. 8121–8125 (2013).
- [6] X. Anguera, L. J. Rodríguez-Fuentes, I. Szöke, A. Buzo, F. Metze, and M. Peñagarikano. Query-by-example spoken term detection on multilingual unconstrained speech. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore*, pp. 2459–2463 (Sept, 2014).
- [7] N. Alcaraz Meseguer. Speech analysis for automatic speech recognition. Master’s thesis, Norwegian University of Science and Technology (2009).

- [8] M. R. Hasan, M. Jamil, M. Rabbani, and M. Rahman. Speaker identification using mel frequency cepstral coefficients. In *3rd International Conference on Electrical and Computer Engineering, Dhaka, Bangladesh*,, vol. 1, pp. 565–568 (2004).
- [9] M. Weintraub. LVCSR log-likelihood ratio scoring for keyword spotting. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-95, Detroit, Michigan, USA*, vol. 1, pp. 297–300 vol.1 (May, 1995). doi: 10.1109/ICASSP.1995.479532.
- [10] K. Ng and V. W. Zue, Subword-based approaches for spoken document retrieval, *Speech Commun.* **32**(3), 157–186 (Oct., 2000). ISSN 0167-6393. doi: 10.1016/S0167-6393(00)00008-X.
- [11] R. Rose and D. Paul. A Hidden Markov Model based keyword recognition system. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-90, Albuquerque, New Mexico, USA*, pp. 129–132 vol.1 (Apr, 1990). doi: 10.1109/ICASSP.1990.115555.
- [12] T. Hazen, W. Shen, and C. White. Query-by-example spoken term detection using phonetic posteriogram templates. In *IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU Merano/Meran, Italy*, pp. 421–426 (Dec, 2009). doi: 10.1109/ASRU.2009.5372889.
- [13] X. Anguera. Speaker independent discriminant feature extraction for acoustic pattern-matching. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Kyoto, Japan*,, pp. 485–488 (March, 2012). doi: 10.1109/ICASSP.2012.6287922.
- [14] Y. Zhang and J. R. Glass. Unsupervised spoken keyword spotting via segmental DTW on Gaussian posteriograms. In *IEEE Workshop on Automatic Speech Recognition & Understanding, ASRU Merano/Meran, Italy*, pp. 398–403 (Dec, 2009). doi: 10.1109/ASRU.2009.5372931.
- [15] H. Wang, C.-C. Leung, T. Lee, B. Ma, and H. Li. An acoustic segment modeling approach to query-by-example spoken term detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Kyoto, Japan*, pp. 5157–5160 (March, 2012). doi: 10.1109/ICASSP.2012.6289081.
- [16] R. R. Pappagari, S. Nayak, and K. S. R. Murty. Unsupervised spoken word retrieval using Gaussian-Bernoulli restricted Boltzmann machines. In *INTERSPEECH 2014, 15th Annual Conference of the International Speech Communication Association, Singapore*, pp. 1737–1741 (Sept, 2014).
- [17] I. Szöke, P. Schwarz, P. Matejka, L. Burget, M. Karafiát, M. Fapso, and J. Černocký. Comparison of keyword spotting approaches for informal continuous speech. In *INTERSPEECH 2005 - Eurospeech, 9th European Conference on Speech Communication and Technology, Lisbon, Portugal*, pp. 633–636 (Sept, 2005).
- [18] C. Chelba, T. J. Hazen, and M. Saralar, Retrieval and browsing of spoken content, *IEEE Signal Processing Mag.* **25**(3), 39–49 (2008).
- [19] M. Saraclar and R. Sproat. Lattice-Based Search for Spoken Utterance Retrieval. In *North American Chapter of the Association for Computational Linguistics, Boston, Massachusetts*, pp. 129–136 (2004).

- [20] L. Mangu, E. Brill, and A. Stolcke, Finding consensus in speech recognition: word error minimization and other applications of confusion networks, *Computer Speech and Language*. **14**(4), 373 – 400 (2000). doi: <http://dx.doi.org/10.1006/csla.2000.0152>.
- [21] D. Hakkani-Tur and G. Riccardi. A general algorithm for word graph matrix decomposition. In *IEEE International Conference on Acoustics, Speech, and Signal Processing, Proceedings. (ICASSP '03), Hong Kong*, vol. 1, pp. I-596–I-599 vol.1 (April, 2003). doi: 10.1109/ICASSP.2003.1198851.
- [22] C. Chelba and A. Acero. Position specific posterior lattices for indexing speech. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, Ann Arbor, Michigan, USA*, pp. 443–450 (June, 2005).
- [23] D. R. H. Miller, M. Kleber, C. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish. Rapid and accurate spoken term detection. In *INTERSPEECH 2007, 8th Annual Conference of the International Speech Communication Association, Antwerp, Belgium*, pp. 314–317 (August, 2007).
- [24] P. Yu, K. Chen, C. Ma, and F. Seide, Vocabulary-independent indexing of spontaneous speech, *IEEE Transactions on Speech and Audio Processing*. **13** (5), 635–643 (Sept, 2005). ISSN 1063-6676. doi: 10.1109/TSA.2005.851881.
- [25] D. Can, E. Cooper, A. Sethy, C. White, B. Ramabhadran, and M. Saracclar. Effect of pronounciations on OOV queries in spoken term detection. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, Taipei, Taiwan*, pp. 3957–3960 (2009).
- [26] T. Ge and Z. Li, Approximate substring matching over uncertain strings, *Proceedings of the VLDB Endowment*. **4**(11), 772–782 (2011).
- [27] G. Aradilla, J. Vepa, and H. Bourlard. Using posterior-based features in template matching for speech recognition. In *International Conference on Spoken Language Processing, Pittsburgh, PA, USA* (2006).
- [28] P. Fousek and H. Hermansky. Towards ASR based on hierarchical posterior-based keyword recognition. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2006, Toulouse, France*, vol. 1, pp. I–I (May, 2006). doi: 10.1109/ICASSP.2006.1660050.
- [29] Y. Zhang, R. Salakhutdinov, H.-A. Chang, and J. Glass. Resource configurable spoken query detection using deep Boltzmann machines. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Kyoto, Japan*, pp. 5161–5164 (2012).
- [30] W. Shen, C. M. White, and T. J. Hazen. A comparison of query-by-example methods for spoken term detection. Technical report, DTIC Document (2009).
- [31] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In eds. U. M. Fayyad and R. Uthurusamy, *KDD Workshop*, pp. 359–370, AAAI Press (1994). ISBN 0-929280-73-3.
- [32] A. Park and J. Glass, Unsupervised pattern discovery in speech, *IEEE Transactions on Audio, Speech, and Language Processing*. **16**(1), 186–197 (Jan, 2008). ISSN 1558-7916. doi: 10.1109/TASL.2007.909282.

- [33] M. Müller, *Information Retrieval for Music and Motion*. Springer-Verlag New York, Inc., Secaucus, NJ, USA (2007). ISBN 3540740473.
- [34] X. Anguera, R. Macrae, and N. Oliver. Partial sequence matching using an unbounded dynamic time warping algorithm. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP, Sheraton Dallas Hotel, Dallas, Texas, USA*, pp. 3582–3585 (March, 2010). doi: 10.1109/ICASSP.2010.5495917.
- [35] M. ANGUERA. Method and system for improved pattern matching (June 25, 2014). EP Patent App. EP20,120,382,508.
- [36] G. Mantena and X. Anguera. Speed improvements to information retrieval-based dynamic time warping using hierarchical k-means clustering. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013, Vancouver, British Columbia, Canada*, pp. 8515–8519 (2013).
- [37] Y. Lin, T. Jiang, and K. Chao, Efficient algorithms for locating the length-constrained heaviest segments with applications to biomolecular sequence analysis, *J. Comput. Syst. Sci.* **65**(3), 570–586 (2002). doi: 10.1016/S0022-0000(02)00010-7.
- [38] Y. Zhang, K. Adl, and J. R. Glass. Fast spoken query detection using lower-bound dynamic time warping on graphical processing units. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP, Kyoto, Japan*, pp. 5173–5176 (March, 2012). doi: 10.1109/ICASSP.2012.6289085.
- [39] Y. Zhang and J. R. Glass. A piecewise aggregate approximation lower-bound estimate for posteriogram-based dynamic time warping. In *INTERSPEECH 2011, 12th Annual Conference of the International Speech Communication Association, Florence, Italy*, pp. 1909–1912 (August, 2011).
- [40] Y. Zhang and J. R. Glass. An inner-product lower-bound estimate for dynamic time warping. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2011, Prague, Czech Republic*, pp. 5660–5663 (2011).
- [41] M. Hochberg, S. Renals, A. Robinson, and G. Cook. Recent improvements to the abbot large vocabulary csr system. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Detroit, Michigan, USA*, vol. 1, pp. 69–72 vol.1 (May, 1995). doi: 10.1109/ICASSP.1995.479275.
- [42] H. Bourlard and N. Morgan. Hybrid hmm/ann systems for speech recognition: Overview and new research directions. In *Adaptive Processing of Sequences and Data Structures*, pp. 389–417. Springer (1998).
- [43] L. Rabiner, A tutorial on Hidden Markov Models and selected applications in speech recognition, *Proceedings of the IEEE*. **77**(2), 257–286 (Feb, 1989). ISSN 0018-9219. doi: 10.1109/5.18626.
- [44] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd edn. Prentice Hall PTR, Upper Saddle River, NJ, USA (1998). ISBN 0132733501.
- [45] K. Rout, R. R. Pappagari, and K. S. R. Murty. Experimental studies on effect of speaking mode on spoken term detection. In *National Conference on Communications 2015 (NCC-2015)*, Mumbai, India (Feb., 2015).

- [46] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*. John Wiley & Sons (2012).
- [47] K. R. Raghavendra Reddy Pappagari and K. S. R. Murty. Query word retrieval from continuous speech using GMM posteriograms. In *International Conference on Signal Processing and Communications (SPCOM), 2014, Bangalore, India*, pp. 1–6 (July, 2014). doi: 10.1109/SPCOM.2014.6984011.
- [48] G. E. Hinton and R. R. Salakhutdinov, Reducing the dimensionality of data with neural networks, *Science*. **313**(5786), 504–507 (July, 2006). doi: 10.1126/science.1127647.
- [49] M. A. Carreira-Perpinan and G. E. Hinton. On contrastive divergence learning. In *Proceedings of the tenth international workshop on artificial intelligence and statistics*, pp. 33–40 (2005).

Chapter 20

Tongue Pattern Recognition to Detect Diabetes Mellitus and Non-Proliferative Diabetic Retinopathy

Bob Zhang

Department of Computer and Information Science

University of Macau, Taipa, Macau

BobZhang@umac.mo

Diabetes Mellitus (DM) and its complications leading to Diabetic Retinopathy (DR) are soon to become one of the 21st century's major health problems. This represents a huge financial burden to health-care officials and governments. To combat this approaching epidemic, this chapter proposes a non-invasive method to detect DM and Non-proliferative Diabetic Retinopathy (NPDR) the initial stage of DR based on three groups of features extracted from tongue images. They include color, texture and geometry. A non-invasive capture device with image correction first captures the tongue images. A tongue color gamut is established with 12 colors representing the tongue color features. The texture values of 8 blocks strategically located on the tongue surface, with the additional mean of all 8 blocks is used to characterize the 9 tongue texture features. Lastly, 13 features extracted from tongue images based on measurements, distances, areas, and their ratios represent the geometry features. Applying a combination of the 34 features, the proposed method can separate Healthy/DM tongues as well as NPDR/DM-sans NPDR (DM samples without NPDR) tongues using features from each of the three groups with average accuracies of 80.52% and 80.33%, respectively. This is on a database consisting of 130 Healthy and 296 DM samples, where 29 of those in DM are NPDR.

20.1. Introduction

World Health Organization (WHO) estimated that in 2000 there were 171 million people worldwide with Diabetes Mellitus (DM), and the number will increase to 366 million by 2030 [1] making the disease among the leading causes of death, disabilities, and economic hardship in the world. Two main types of DM exist, Type 1 DM and Type 2 DM. People with Type 1 DM fail

to produce insulin, and therefore require injections of it. Type 2 DM is the most common type and can be categorized by insulin resistance. Currently, there is no cure for Type 1 DM or Type 2 DM. However, Type 2 DM can be managed by eating well, exercising, and maintaining a healthy lifestyle.

A fasting plasma glucose (FPG) test is the standard method practiced by many medical professionals to diagnose DM. FPG test is performed after the patient has gone at least 12 hours without food, and requires taking a sample of the patient's blood (by piercing their finger) in order to analyze its blood glucose levels. Even though this method is accurate, it can be considered invasive, and slightly painful (piercing process). Diabetic Retinopathy (DR) is a microvascular complication of DM that is responsible for 4.8% of the 37 million cases of blindness in the world, estimated by WHO¹. In its earliest stage known as Non-proliferative Diabetic Retinopathy (NPDR), the disease if detected can be treated to prevent further progression and sight loss. Various imaging modalities such as red-free [2, 3], angiography [3, 4], and color fundus imaging [5–10] are used to examine the human retina in order to detect DR and subsequently NPDR. These methods are based on the detection of relevant features related to DR, including but not limited to hemorrhages, microaneurysms, various exudates, and retinal blood vessels. These imaging modalities themselves can be regarded as invasive, exposing the eye to bright flashes or having fluorescein injected into a vein in the case of angiography. Therefore, there is a need to develop a non-invasive yet accurate DM and NPDR detection method.

As a result, this chapter deals with the above-mentioned problems and proposes a non-invasive automated method to detect DM and NPDR by distinguishing Healthy/DM and NPDR/DM-sans NPDR (DM without NPDR) samples using an array of tongue features consisting of color, texture, and geometry. The human tongue contains numerous features that can be used to diagnose disease [11–15], with color, texture, geometry features being the most prominent [11–15]. Traditionally, medical practitioners would examine these features based on years of experience [11–25]. However, ambiguity and subjectivity are always associated with their diagnostic results. To remove these qualitative aspects, quantitative feature extraction and analysis from tongue images can be established. To the best of our knowledge, there is no other published work to detect DM or NPDR using tongue color, texture, and geometry features.

Tongue images were captured using a specially designed in-house device taking into consideration color correction [26]. Each image was segmented [27] in order to locate its foreground pixels. With the relevant pixels

located, three groups of features namely color, texture, and geometry were extracted from the tongue foreground. To conduct experimental results a dataset consisting of 130 Healthy samples taken from Guangdong Provincial Hospital of Traditional Chinese Medicine, Guangdong, China, and 296 DM samples consisting of 267 DM-sans NPDR, and 29 NPDR processed from Hong Kong Foundation for Research and Development in Diabetes, Prince of Wales Hospital, Hong Kong SAR were used. Classification was performed between Healthy vs. DM in addition to NPDR vs. DM-sans NPDR initially using every feature individually (from the three groups), followed by an optimal combination of all the features.

The rest of this chapter is organized as follows. Section 20.2 describes the tongue image capture device, color correction, and tongue segmentation, while Section 20.3 discusses tongue color feature extraction. In Section 20.4, tongue texture feature extraction is given in detail, with tongue geometry feature extraction presented in Section 20.5. Section 20.6 describes the experimental results and discussion, followed by concluding remarks in Section 20.7.

20.2. Capture Device and Tongue Image Processing

The capture device, color correction of the tongue images, and tongue segmentation are given in this section. Figure 20.1 shows the in-house designed device consisting of a three chip CCD camera with 8 bit resolution, and two D65 fluorescent tubes placed symmetrically around the camera in order to produce a uniform illumination. The angle between the incident light and emergent light is 45° , recommended by Commission Internationale de l'Eclairage (CIE). During image capture patients placed their chin on a chinrest while showing their tongue to the camera. The images captured in JPEG format ranged from 257×189 pixels to 443×355 pixels were color corrected [26] to eliminate any variability in color images caused by changes of illumination and device dependency. This allows for consistent feature extraction and classification in the following steps. The idea of [26] (based on the Munsell ColorChecker) is to map a matrix generated from the input RGB vector to an objective sRGB vector, thereby obtaining a transformation model. Compared with the retinal imaging modalities mentioned above, this capture device is non-invasive, neither requiring a bright flash nor injection of dye into a patient's blood stream.

Once the tongue images are captured, automatic segmentation [27] is next applied to each image in order to separate its foreground pixels from its

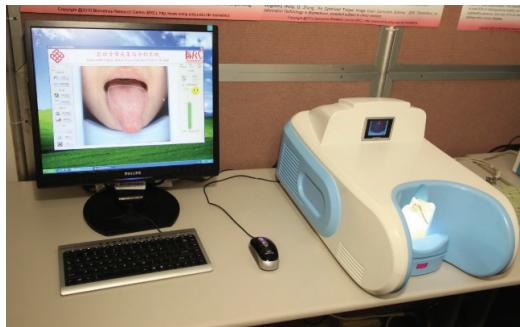


Fig. 20.1. Tongue capture device.

background. This is accomplished by combining a bi-elliptical deformable template (BEDT), and an active contour model known as bi-elliptical deformable contour (BEDC). In [27] the segmented tongue is obtained by first minimizing the energy function of BEDT, followed by the energy function of BEDC. BEDT captures the overall tongue shape features, while BEDC can be deformed to match the local tongue details. The result is a binary tongue image clearly defining foreground pixels (tongue surface area and its edges) from its background pixels (area outside the tongue edges). This allows for three groups of features, color, texture, and geometry to be extracted from a tongue foreground image in the proceeding steps.

20.3. Tongue Color Features

The following section describes how color features are extracted from tongue images. The tongue color gamut is first summarized in Section 20.3.1. In Section 20.3.2, every foreground tongue pixel is compared to 12 colors representing the tongue color gamut and assigned its nearest color. This forms the color features.

20.3.1. Tongue color gamut

The tongue color gamut [28] represents all possible colors that appear on the tongue surface, and exists within the red boundary shown in Fig. 20.2. It was created by plotting each tongue foreground pixel in our dataset onto the CIE 1931 chromaticity diagram (refer to Fig. 20.2), which shows all possible colors in the visible spectrum. Further investigation revealed that 98% of the tongue pixels lie inside the black boundary. To better represent the tongue color gamut, 12 colors plotted in Fig. 20.3 were selected with the

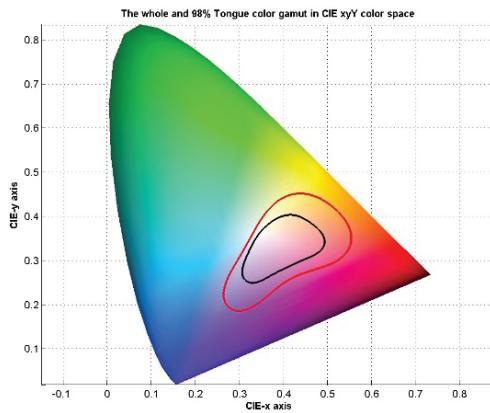


Fig. 20.2. A color gamut in the CIExyY color space depicting the tongue color gamut inside the red boundary. Furthermore, 98% of the tongue color gamut can be located within the black boundary.

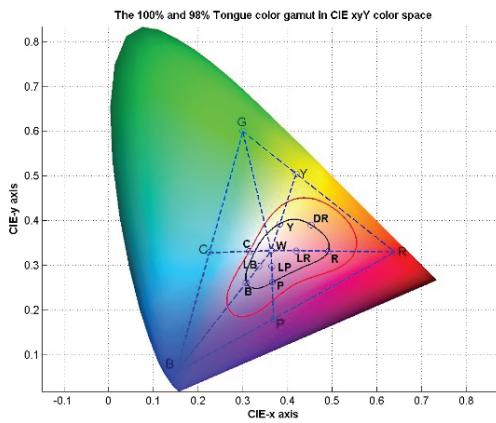


Fig. 20.3. The tongue color gamut can be represented using several points by drawing lines from the RGB color space.

help of the RGB color space. On the RG line a point Y (Yellow) is marked. Between RB a point P (Purple) is marked, and C (Cyan) is marked between GB. The center of the RGB color space is calculated and designated as W (White), the first of the 12 colors (see Fig. 20.3). Then, for each R (Red), B (Blue), Y, P, and C point, a straight line is drawn to W. Each time these

lines intersect the tongue color gamut, a new color is added to represent the 12 colors. This accounts for R, Y, C, B and P. LR (Light red), LP (Light purple), and LB (Light blue) are midpoints between lines from the black boundary to W, while DR (Deep red) is selected as no previous point occupies that area. More details about the tongue color gamut can be found in [28]. GY (Gray) and BK (Black) are not shown in Fig. 20.3 since both belong to grayscale.

The 12 colors representing the tongue color gamut are extracted from Fig. 20.3 and shown in Fig. 20.4 as a color square with its label on top. Correspondingly, its RGB and CIELAB values are given in Table 20.1.

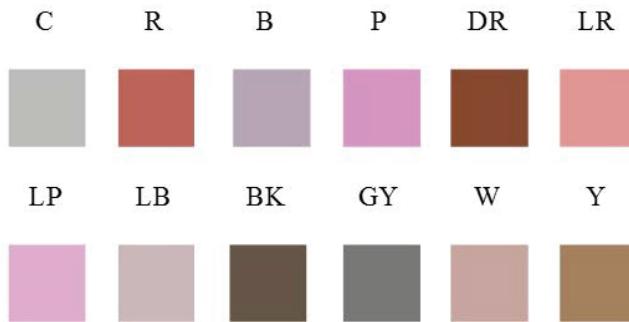


Fig. 20.4. 12 colors representing the tongue color gamut with its label on top.

Table 20.1. RGB and CIELAB values of the 12 colors.

Color	[R G B]	[L A B]
C (Cyan)	[188 188 185]	[76.0693 -0.5580 1.3615]
R (Red)	[189 99 91]	[52.2540 34.8412 21.3002]
B (Blue)	[183 165 180]	[69.4695 9.5423 -5.4951]
P (Purple)	[226 142 214]	[69.4695 42.4732 -23.8880]
DR (Deep red)	[136 72 49]	[37.8424 24.5503 25.9396]
LR (Light red)	[227 150 147]	[69.4695 28.4947 13.3940]
LP (Light purple)	[225 173 207]	[76.0693 24.3246 -9.7749]
LB (Light blue)	[204 183 186]	[76.0693 7.8917 0.9885]
BK (Black)	[107 86 56]	[37.8424 3.9632 20.5874]
GY (Gray)	[163 146 143]	[61.6542 5.7160 3.7317]
W (White)	[200 167 160]	[70.9763 10.9843 8.2952]
Y (Yellow)	[166 129 93]	[56.3164 9.5539 24.4546]

20.3.2. Color feature extraction

For the foreground pixels of a tongue image, corresponding RGB values are first extracted, and converted to CIELAB [29] by transferring RBG to

CIEXYZ using (20.1):

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.4124 & 0.3576 & 0.1805 \\ 0.2126 & 0.7152 & 0.0722 \\ 0.0193 & 0.1192 & 0.9505 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \quad (20.1)$$

followed by CIEXYZ to CIELAB via (20.2):

$$\begin{cases} L^* = 166 \cdot f(Y/Y_0) - 16 \\ a^* = 500 \cdot [f(X/X_0) - f(Y/Y_0)] \\ b^* = 200 \cdot [f(Y/Y_0) - f(Z/Z_0)] \end{cases} \quad (20.2)$$

where $f(x) = x^{1/3}$ if $x > 0.008856$ or $f(x) = 7.787x + 16/116$ if $x \leq 0.008856$.

X_0 , Y_0 , and Z_0 in (20.2) are the CIEXYZ tristimulus values of the reference white point. The LAB values are then compared to 12 colors from the tongue color gamut (see Table 20.1) and assigned the color which is closest to it (measured using Euclidean distance). After evaluating all tongue foreground pixels, the total of each color is summed and divided by the total number of pixels. This ratio of the 12 colors forms the tongue color feature vector v , where $v = [c_1, c_2, c_3, c_4, c_5, c_6, c_7, c_8, c_9, c_{10}, c_{11}, c_{12}]$ and c_i represents the sequence of colors in Table 20.1. As an example, the color features of three tongues are shown in visual form (refer to Figs. 20.5–20.7) along with its extracted color feature vector, where the original image is

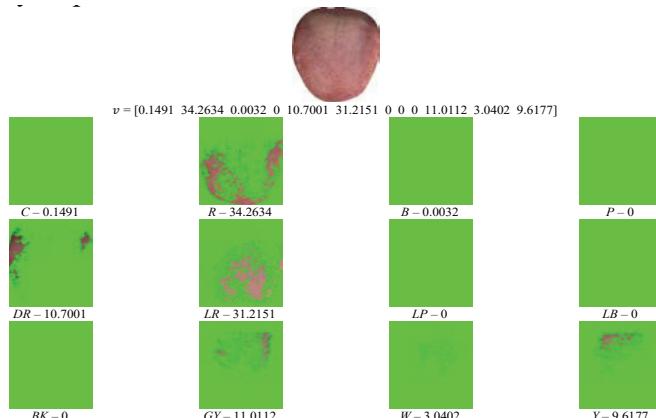


Fig. 20.5. Healthy tongue sample, its tongue color feature vector and corresponding 12 color makeup with most of the pixels classified as R.

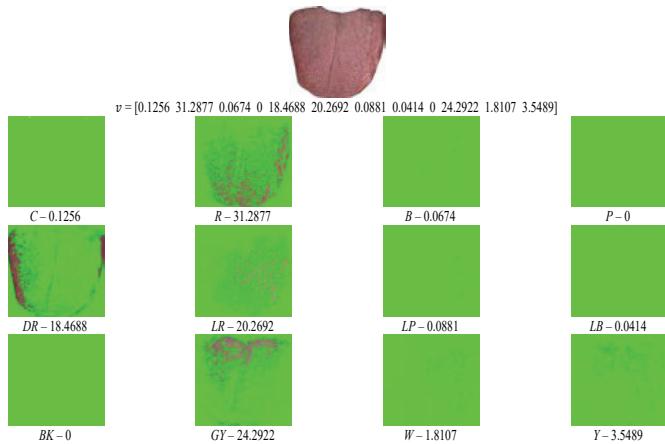


Fig. 20.6. DM tongue sample, its tongue color feature vector and corresponding 12 color makeup with most of the pixels classified as R.

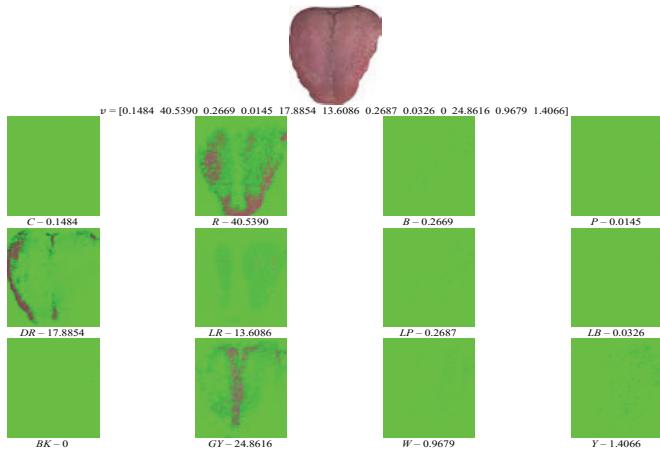


Fig. 20.7. NPDR tongue sample, its tongue color feature vector and corresponding 12 color makeup with most of the pixels classified as R.

decomposed into one of the 12 colors. Figure 20.5 is from a Healthy sample, Fig. 20.6 is a DM sample, while a NPDR sample is given in Fig. 20.7. In these three samples the majority of pixels are R.

The mean colors of Healthy, DM, and NPDR are displayed in Table 20.2 along with their standard deviation (std). DM tongues have a higher ratio

Table 20.2. *Mean (std)* of the color features for Healthy ($std_{\text{Healthy}} = 11.71$), DM ($std_{\text{DM}} = 12.50$), and NPDR ($std_{\text{NPDR}} = 12.07$).

	C	R	DR	LR	GY	W	Y
Healthy	23.77 (3.76)	30.94 (10.73)	12.71 (8.11)	14.12 (9.36)	10.54 (10.80)	1.3 (1.84)	6.53 (6.35)
	24.80 (4.84)	30.70 (11.47)	18.40 (12.68)	10.53 (8.73)	10.80 (12.03)	1.07 (1.79)	3.55 (5.62)
DM	24.14 (4.86)	28.54 (13.13)	14.31 (10.38)	11.12 (7.74)	15.50 (13.92)	1.73 (2.16)	4.48 (6.82)

of *DR*, *LR* and *Y* are greater in Healthy samples, and *GY* is higher in NPDR. The rest of the mean color features are similar. Only 7 colors are listed out of the 12 as the remaining 5 have ratios less than 1%.

20.4. Tongue Texture Features

Texture feature extraction from tongue images is presented in this section. To better represent the texture of tongue images, 8 blocks (see Fig. 20.8) of size 64×64 strategically located on the tongue surface are used. A block size of 64×64 was chosen due to the fact that it covers all 8 surface areas very well, while achieving minimum overlap. Larger blocks would cover areas outside the tongue boundary, and overlap more with other blocks. Smaller block sizes would prevent overlapping, but not cover the 8 areas as efficiently. The blocks are calculated automatically by first locating the center of the tongue using a segmented binary tongue foreground image. Following this, the edges of the tongue are established and equal parts are measured from its center to position the 8 blocks. Block 1 is located at the tip; Blocks 2 and 3, and Blocks 4 and 5 are on either side; Blocks 6 and 7 are at the root, and Block 8 is at the center.

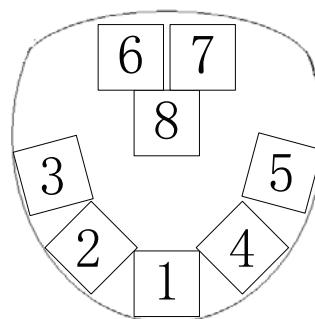


Fig. 20.8. Location of the 8 texture blocks on the tongue.

The Gabor filter is a linear filter used in image processing, and is commonly used in texture representation. To compute the texture value of each block, the 2-dimensional Gabor filter is applied and defined as:

$$G_k(x, y) = \exp\left(\frac{x'^2 + y^2 y'^2}{-2\sigma^2}\right) \cos\left(2\pi \frac{x'}{\lambda}\right) \quad (20.3)$$

where $x' = x \cos \theta + y \sin \theta$, $y' = -x \sin \theta + y \cos \theta$, σ is the variance, λ is the wavelength and γ is the aspect ratio of the sinusoidal function, and θ is the orientation. A total of three σ (1, 2 and 3) and four θ (0° , 45° , 90° , and 135°) choices were investigated to achieve the best result. Each filter is convolved with a texture block to produce a response $R_k(x, y)$:

$$R_k(x, y) = G_k(x, y) * im(x, y) \quad (20.4)$$

where $im(x, y)$ is the texture block and $*$ represents 2-D convolution. Responses of a block are combined to form FR_i , its final response evaluated as:

$$FR_i(x, y) = \max(R_1(x, y), R_2(x, y), \dots, R_n(x, y)) \quad (20.5)$$

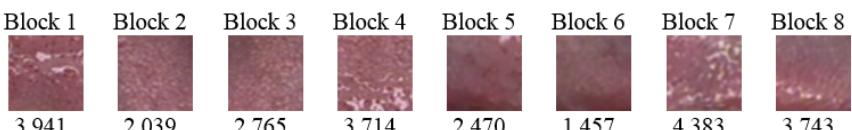
which selects the maximum pixel intensities, and represents the texture of a block by averaging the pixel values of FR_i .



(a) Healthy texture blocks



(b) DM texture blocks



(c) NPDR texture blocks

Fig. 20.9. Texture blocks with their texture values below them.

In the end σ equal to 1 and 2 with three orientations (45° , 90° , and 135°) was chosen. This is due to the fact that the sum of all texture blocks between Healthy and DM had the largest absolute difference. Figure 20.9 illustrates the texture blocks for Healthy, DM, and NPDR samples. Below each block its corresponding texture value is provided.

Table 20.3 depicts the texture value mean for Healthy, DM, and NPDR together with their standard deviation. Healthy samples have a higher texture value in Block 7, whereas NPDR texture values are greater for the remaining blocks. An additional texture value, the mean of all 8 blocks is also included. This brings the total number of texture features extracted from tongue images to be 9.

Table 20.3. *Mean (std)* of the geometry features for Healthy ($std_{\text{Healthy}} = 33013.78$), DM ($std_{\text{DM}} = 33723.85$), and NPDR ($std_{\text{NPDR}} = 35673.93$).

	Healthy	DM	NPDR
w	320.8077 (36.5289)	335.9189 (42.7073)	344.3103 (39.4788)
l	302.6231 (43.4015)	295.2703 (62.2306)	309.8621 (64.9713)
lw	0.9527 (0.1638)	0.8897 (0.2067)	0.9117 (0.2202)
z	144.5385 (16.6250)	141.1875 (25.1676)	146.069 (23.6574)
cd	-49.6231 (29.2308)	-66.6926 (30.9031)	-64.6552 (34.3067)
cdr	-0.1631 (0.0871)	-0.2249 (0.0900)	-0.2097 (0.1029)
a	76709.14 (15525.3172)	76961.31 (21599.4127)	83286.67 (22629.9217)
ca	66493.77 (15079.8031)	64607.43 (21983.2771)	68727.14 (20900.1437)
car	0.8635 (0.0873)	0.8232 (0.1232)	0.8155 (0.1153)
sa	84662.52 (19200.1304)	82260.71 (27989.9621)	87506.07 (26610.9335)
sar	0.8908 (0.0703)	0.871689 (0.0848)	0.886897 (0.0920)
ta	32092.11 (7336.0657)	36077.43 (10624.3571)	37959.16 (9973.8946)
tar	0.4212 (0.0631)	0.4722 (0.0745)	0.4624 (0.0807)

20.5. Tongue Geometry Features

In the following section we describe 13 geometry features extracted from tongue images. These features are based on measurements, distances, areas, and their ratios.

Width

The width (w) feature (see Fig. 20.10) is measured as the horizontal distance along the x -axis from a tongue's furthest right edge point (x_{max}) to its furthest left edge point (x_{min}):

$$w = x_{max} - x_{min}. \quad (20.6)$$

Length

The length (l) feature (see Fig. 20.10) is measured as the vertical distance along the y -axis from a tongue's furthest bottom edge point (y_{max}) to its furthest top edge point (y_{min}):

$$l = y_{max} - y_{min}. \quad (20.7)$$

Length-Width Ratio

The length-width ratio (lw) is the ratio of a tongue's length to its width:

$$lw = l/w. \quad (20.8)$$

Smaller-Half-Distance

Smaller-half-distance (z) is the half distance of l or w depending on which segment is shorter (see Fig. 20.10):

$$z = \min(l, w)/2. \quad (20.9)$$

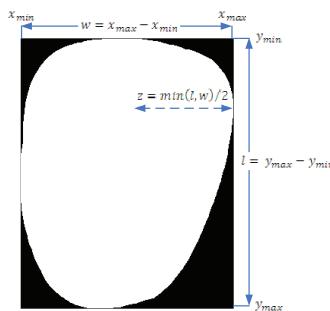


Fig. 20.10. Illustration of features 1, 2, and 4.

Center Distance

The center distance (cd) (refer to Fig. 20.11) is distance from w 's y -axis center point to the center point of $l(y_{cp})$:

$$cd = \frac{(\max(y_{x_{max}}) + \max(y_{x_{min}}))}{2} - y_{cp}, \quad (20.10)$$

where $y_{cp} = (y_{max} + y_{min})/2$.

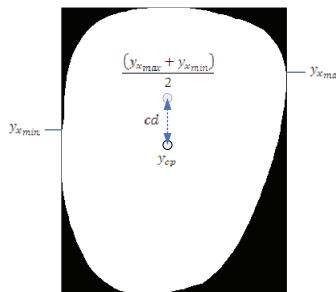


Fig. 20.11. Illustration of feature 5.

Center Distance Ratio

Center distance ratio (cdr) is ratio of cd to l :

$$cdr = \frac{cd}{l}. \quad (20.11)$$

Area

The area (a) of a tongue is defined as the number of tongue foreground pixels.

Circle Area

Circle area (ca) is the area of a circle within the tongue foreground using smaller-half-distance z , where $r = z$ (refer to Fig. 20.12):

$$ca = \pi r^2. \quad (20.12)$$

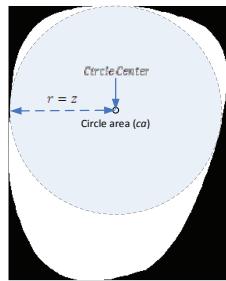


Fig. 20.12. Illustration of feature 8.

Circle Area Ratio

Circle area ratio (car) is the ratio of ca to a :

$$car = \frac{ca}{a}. \quad (20.13)$$

Square Area

Square area (sa) is the area of a square defined within the tongue foreground using smaller-half-distance z (refer to Fig. 20.13):

$$sa = 4z^2. \quad (20.14)$$

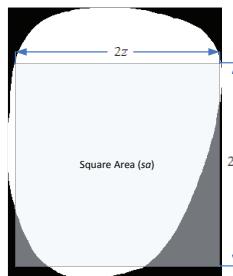


Fig. 20.13. Illustration of feature 10.

Square Area Ratio

Square area ratio (sar) is the ratio of sa to a :

$$sar = \frac{sa}{a}. \quad (20.15)$$

Triangle Area

Triangle area (ta) is the area of a triangle defined within the tongue foreground (see Fig. 20.14). The right point of the triangle is x_{max} , the left point is x_{min} , and the bottom is y_{max} .

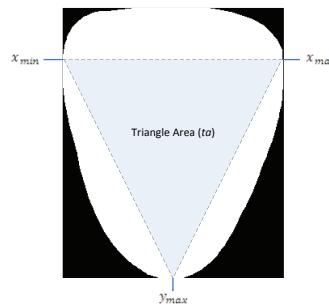


Fig. 20.14. Illustration of feature 12.

Triangle Area Ratio

Triangle area ratio (tar) is the ratio of ta to a :

$$tar = \frac{ta}{a} \quad (20.16)$$

The mean geometry features of Healthy, DM, and NPDR are shown in Table 20.3 along with their standard deviations.

20.6. Numerical Results and Discussions

The ensuing section presents the numerical results. Healthy vs. DM classification is first provided in Section 20.6.1. This is followed by NPDR vs. DM-sans NPDR classification in Section 20.6.2.

20.6.1. *Healthy vs. DM classification*

The numerical results were obtained on the tongue image database comprised of 426 images divided into 130 Healthy, and 296 DM (refer to Section 20.1). Healthy samples were verified through a blood test and other examinations. If indicators from these tests fall within a certain range they were deemed healthy. In the DM class FPG test was used to diagnose diabetes.

Half the images were randomly selected for training, while the other half was used as testing. This process was repeated five times. Classification was performed using k -NN [30] (with $k = 1$) and SVM [31], where the kernel function (linear) mapped the training data into kernel space. To measure the performance average accuracy was employed:

$$\text{Average Accuracy} = \frac{\text{sensitivity} + \text{specificity}}{2}. \quad (20.17)$$

with the average of all five repetitions recorded as the final classification rate. In the first step, each individual feature (from the three groups) was applied to discriminate Healthy vs. DM. This result can be seen in Tables 20.4–20.6. It should be noted that both k -NN and SVM achieved the same average accuracy for all 34 features. The highest average accuracy of 66.26% from this step was obtained using the geometry feature *ta* (refer to Table 20.6).

Table 20.4. Classification result of k -NN and SVM using each color individually to discriminate Healthy vs. DM.

Feature Number	Feature Name	Average Accuracy (%)
1	C	52.17
2	R	48.38
3	B	55.24
4	P	43.90
5	DR	60.16
6	LR	57.44
7	LP	54.30
8	LB	51.98
9	BK	50.00
10	GY	48.55
11	W	51.80
12	Y	64.78

In the next step, optimization by feature selection using sequential forward selection (SFS) was performed. SFS is a feature selection method that begins with an empty set of features. It adds additional features based on maximizing some criterion J , and terminates when all features have been added. In our case J is the average accuracy of the classifier (k -NN and SVM). Tables 20.7 (k -NN) and 20.8 (SVM) illustrate this result applied to each of the three main feature groups. From color features the best combination is 3 and 12, which obtained an average accuracy of 68.76% using SVM (see Table 20.8). In texture features, 14, 15, 16, and 17 attained an

Table 20.5. Classification result of k -NN and SVM using each texture block individually to discriminate Healthy vs. DM.

Feature Number	Feature Name	Average Accuracy (%)
13	Block 1	53.71
14	Block 2	62.66
15	Block 3	61.30
16	Block 4	56.63
17	Block 5	56.50
18	Block 6	48.58
19	Block 7	54.59
20	Block 8	54.17
21	Mean of Blocks 1-8	51.15

Table 20.6. Classification result of k -NN and SVM using each geometry feature individually to discriminate Healthy vs. DM.

Feature Number	Feature Name	Average Accuracy (%)
22	w	60.81
23	l	50.25
24	lw	58.44
25	z	53.33
26	cd	60.29
27	cdr	61.61
28	a	43.76
29	ca	55.02
30	car	59.15
31	sa	55.02
32	sar	54.76
33	ta	66.26
34	tar	64.68

Table 20.7. Optimization of Healthy vs. DM classification using feature selection with k -NN.

Grouping	Feature(s) Number(s)	Feature(s) Name(s)	Average Accuracy (%)
Color	12	Y	64.78
Texture	14,17, 16, 19	Blocks 2, 4, 5, 9	67.48
Geometry	22-30, 32-34	$w-car$, $sar-tar$	67.87
Best of Color, Texture, and Geometry	12,14, 16, 17, 19, 22-30, 32-34	Y , Blocks 2, 4, 5, 9, $w-car$, $sar-tar$	67.87
All Features	1-30, 32-34	$C-car$, $sar-tar$	67.87

Table 20.8. Optimization of Healthy vs. DM classification using feature selection with SVM.

Grouping	Feature Numbers	Feature Names	Average Accuracy (%)
Color	3,12	B,Y	68.76
Texture	14-17	Blocks 2-5	67.67
Geometry	22,30, 32-34	w,car, sar-tar	69.09
Best of Color, Texture, and Geometry	3,12,14, 15, 17, 22,30, 32-34	B,Y, Blocks 2, 3, 5,w,car, sar-tar	77.39
All Features	3, 5, 12, 15, 22, 27, 30, 33, 34	B, DR, Y, Block 3,w , cdr ,car, ta, tar	80.52

average accuracy of 67.67%, again using SVM. With geometry features, 22, 30, 32, 33, and 34 distinguished Healthy vs. DM with an average accuracy of 69.09% (in Table 20.8). Combining the features in these three groups by applying SFS, an average accuracy of 77.39% was achieved in Table 20.8 using 3, 12, 14, 15, 17, 22, 30, 32, 33, and 34. Finally, by examining the best combination from all features (SFS), the highest average accuracy of 80.52% can be accomplished (via SVM), with a sensitivity of 90.77% and a specificity of 70.27%. ROC analysis was also performed on this classification as shown by the red ROC curve in Fig. 20.15. The average accuracy of this result is higher than the optimal combination from the three feature groups (77.39%), and contains fewer features. At the same time, it significantly improves upon the use of all features without feature selection,

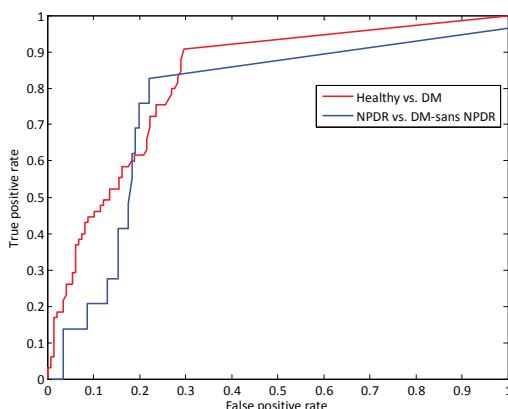


Fig. 20.15. ROC curves for Healthy vs. DM (red), and NPDR vs. DM-sans NPDR (blue).

which obtained an average accuracy of 58.06% (k -NN) and 44.68% (SVM). Figure 20.16 depicts three typical samples from Healthy and DM.

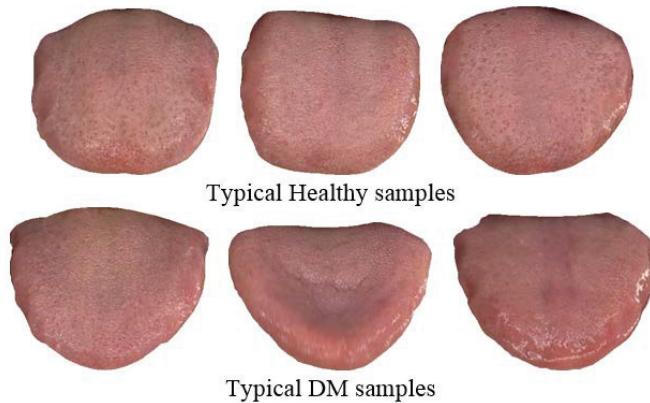


Fig. 20.16. Typical Healthy (top) and DM (bottom) tongue samples.

20.6.2. NPDR vs. DM-sans NPDR classification

Of the 296 DM samples 29 were marked as NPDR (refer to Section 20.1). The NPDR samples were verified by medical professionals after examining the retina of patients. Using the same experimental setup as Section 20.6.1, the results of NPDR vs. DM-sans NPDR (267 samples) classification are illustrated in Tables 20.9–20.12. Since it was established in the preceding

Table 20.9. Classification results of using each color individually to discriminate NPDR vs. DM-sans NPDR using SVM.

Feature Number	Feature Name	Average Accuracy (%)
1	<i>C</i>	54.26
2	<i>R</i>	40.09
3	<i>B</i>	57.24
4	<i>P</i>	50.29
5	<i>DR</i>	61.07
6	<i>LR</i>	52.98
7	<i>LP</i>	45.19
8	<i>LB</i>	55.07
9	<i>BK</i>	49.44
10	<i>GY</i>	70.81
11	<i>W</i>	64.14
12	<i>Y</i>	53.12

Table 20.10. Classification results of using each texture block individually to discriminate NPDR vs. DM-sans NPDR using SVM.

Feature Number	Feature Name	Average Accuracy (%)
13	Block 1	50.36
14	Block 2	58.22
15	Block 3	55.22
16	Block 4	67.07
17	Block 5	61.45
18	Block 6	52.82
19	Block 7	48.41
20	Block 8	49.15
21	Mean of Blocks 1-8	61.00

Table 20.11. Classification results of using each geometry feature individually to discriminate NPDR vs. DM-sans NPDR using SVM.

Feature Number	Feature Name	Average Accuracy (%)
22	w	30.12
23	l	52.98
24	lw	46.08
25	z	56.43
26	cd	43.91
27	cdr	48.48
28	a	62.57
29	ca	58.30
30	car	46.45
31	sa	58.30
32	sar	58.08
33	ta	32.07
34	tar	45.49

Table 20.12. Optimization of NPDR vs. DM-sans NPDR classification using feature selection with SVM.

Grouping	Feature Numbers	Feature Names	Average Accuracy (%)
Color	1, 3, 5, 7-11	$C, B, DR, LP-W$	75.84
Texture	13-17, 21	Blocks 1-5, Mean of Blocks 1-8	72.09
Geometry	23, 28, 32	l, a, sar	65.27
Best of Color, Texture, and Geometry	3, 7, 10, 28	B, LP, GY, a	79.21
All Features	7, 10, 11, 14, 25	$LP, GY, W, Block 2, z$	80.33

section that SVM outperforms k -NN, only the former classifier was used. The average accuracies of applying each feature individually from the three groups are shown in Tables 20.9–20.11, and Table 20.12 displays the optimized result using SFS. As can be seen in the last row of Table 20.12, the best result of 80.33% was achieved with features 7, 10, 11, 14, and 25 (sensitivity - 82.76% and specificity - 77.90% based on its blue ROC curve in Fig. 20.15). This compares to 59.05% and 53.35% average accuracies for k -NN and SVM respectively using all features without feature selection. Mean of the 5 optimal features for DM-sans NPDR and NPDR can be found in Table 20.13 along with its three typical samples (refer to Fig. 20.17).

Table 20.13. Mean of the optimal tongue features for DM-sans NPDR and NPDR.

	10	25	7	14	11
DM-sans NPDR	10.2884	140.6573	0.0647	2.1205	1.0025
NPDR	15.5041	146.0690	0.0565	2.3406	1.7322

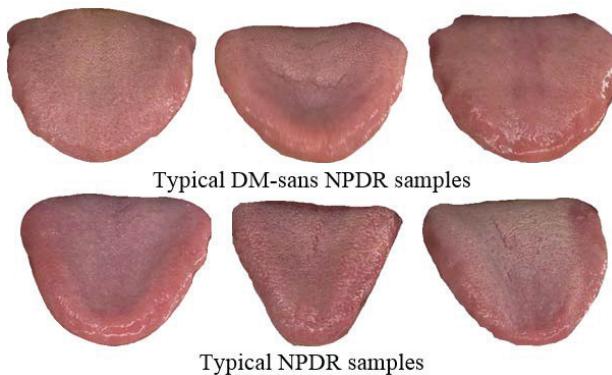


Fig. 20.17. Typical DM-sans NPDR (top) and NPDR (bottom) tongue samples.

For completeness NPDR vs. Healthy classification was also conducted. An average accuracy of 87.14% was accomplished using SFS with SVM, achieving a sensitivity of 89.66% and a specificity of 84.62% via features 3, 9, 15, 16, and 33.

20.7. Conclusions

In this article, a non-invasive approach to classify Healthy/DM and NPDR/DM-sans NPDR samples using three groups of features extracted from tongue images was proposed. These three groups include color, texture, and geometry. A tongue color gamut was first applied such that each tongue image can be represented by 12 colors. Afterwards, 8 blocks strategically located on the tongue were extracted and its texture value calculated. Finally, 13 geometry features from tongue images based on measurements, distances, areas, and their ratios were extracted. Numerical experiments were carried out using 130 Healthy and 296 DM tongue images. By applying each feature individually to separate Healthy/DM, the highest average accuracy achieved (via SVM) was only 66.26%. However, employing SFS with SVM, 9 features (with elements from all the three groups) were shown to produce the optimal result, obtaining an average accuracy of 80.52%. As for NPDR/DM-sans NPDR classification, the best result of 80.33% was attained using 5 features, 3 from color, and one from texture and geometry. This lays the groundwork for a potentially new way to detect DM, while providing a novel means to detect NPDR without retinal imaging or analysis.

Acknowledgments

The author would like to thank the Biometric Research Center at the Hong Kong Polytechnic University for sharing the database. This work is supported by the Science and Technology Development Fund (FDCT) of Macao SAR (FDCT/128/2013/A and FDCT/124/2014/A3).

References

- [1] World Health Organization, *Prevention of blindness from diabetes mellitus*. (2006).
- [2] J.H. Hipwell and F. Strachan, eds., Automated detection of microaneurysms in digital red-free photographs: A diabetic retinopathy screening tool, *Diabet Med.* **7**, 588–594 (2000).
- [3] M.E. Martinez-Perez and A. Hughes, eds., Segmentation of blood vessels from red-free and fluorescein retinal images, *Medical Image Analysis*. **11**, 47–61 (2007).
- [4] T. Spencer and J.A. Olson, eds., An image-processing strategy for the segmentation of microaneurysms in fluorescein angiograms of the ocular fundus, *Computers and Biomedical Research*. **29**, 284–302 (1996).

- [5] M. Niemeijer and B. van Ginneken, eds., Automatic detection of red lesions in digital color fundus photographs, *IEEE Trans. on Medical Imaging*. **24**, 584–592 (2005).
- [6] M. Niemeijer and B. van Ginneken, eds., Retinopathy online challenge: automatic detection of microaneurysms in digital color fundus photographs, *IEEE Trans. on Medical Imaging*. **29**, 185–195 (2010).
- [7] J.J. Staal and M.D. Abramoff, eds., Ridge based vessel segmentation in color images of the retina, *IEEE Trans. on Medical Imaging*. **23**, 501–509 (2004).
- [8] B. Zhang and F. Karray, eds., Sparse Representation Classifier for microaneurysm detection and retinal blood vessel extraction, *Inf. Sci.* **200**, 78–90 (2012).
- [9] B. Zhang and X. Wu, eds., Detection of microaneurysms using multi-scale correlation coefficients, *Pattern Recognition*, **43**, 2237–2248 (2010).
- [10] B. Zhang and L. Zhang, eds., Retinal vessel extraction by matched filter with first-order derivative of Gaussian, *Comp. in Bio. and Med.*, **40**, 438–445 (2010).
- [11] C.C. Chiu, The development of a computerized tongue diagnosis system, *Engineering: Applications, Basis and Communications*. **8**, 342–350 (1996).
- [12] C.C. Chiu, A novel approach based on computerized image analysis for traditional Chinese medical diagnosis of the tongue, *Computer Methods and Programs in Biomedicine*. **61**, 77–89 (2000).
- [13] B. Kirschbaum, *Atlas of Chinese Tongue Diagnosis*. Eastland Press, Seattle, (2000).
- [14] D. Zhang and B. Pang, eds., Computerized diagnosis from tongue appearance using quantitative feature classification, *American Journal of Chinese Medicine*. **33**, 859–866 (2005).
- [15] Y. Zhang and R. Liang, eds., Analysis of the color characteristics of tongue digital images from 884 physical examination cases, *Journal of Beijing University of Traditional Chinese Medicine*. **28**, 73–75 (2005).
- [16] D. Zhang, *Automated Biometrics: Technologies and Systems*. Kluwer Academic Publisher, Boston, (2000).
- [17] W. Su and Z.Y. Xu, eds., Objectified study on tongue images of patients with lung cancer of different syndromes, *Chinese Journal of Integrative Medicine*. **17**, 272–276 (2011).
- [18] K.Q. Wang and D. Zhang, eds., *Tongue diagnosis based on biometric pattern recognition technology*. in Pattern recognition from classical to modern approaches, 1st edition, S.K. Pal, and A. Pal, World Scientific. (2001).
- [19] B. Huang and J. Wu, D. Zhang, eds., Tongue shape classification by geometric features, *Inf. Sci.* **180**, 312–324 (2010).
- [20] B. Li and Q. Huang, eds., A method of classifying tongue colors for traditional chinese medicine diagnosis based on the CIELAB color space. In *Proc. International Conference on Medical Biometrics* 153–159 (2007).
- [21] C.H. Li, and P.C. Yuen, Tongue image matching using color content, *Pattern Recognition*. **35**, 407–419 (2002).
- [22] N.M. Li, *The contemporary investigations of computerized tongue diagnosis, The Handbook of Chinese Tongue Diagnosis*. Shad-Yuan Publishing (1994).

- [23] G. Maciocia, *Tongue Diagnosis in Chinese Medicine*. Eastland Press, Seattle (1995).
- [24] N. Ohta and A.R. Robertson, *Colorimetry: Fundamentals and Applications*. John Wiley & Sons (2006).
- [25] B. Pang and D. Zhang, eds., Tongue image analysis for appendicitis diagnosis, *Inf. Sci.* **17**, 160–176 (2005).
- [26] X. Wang and D. Zhang, An optimized tongue image color correction scheme, *IEEE Trans. on Information Technology in Biomedicine*. **14**, 1355–1364 (2010).
- [27] B. Pang and D. Zhang, eds., The bi-elliptical deformable contour and its application to automated tongue segmentation in Chinese medicine, *IEEE Trans. on Medical Imaging*. **24**, 946–956 (2005).
- [28] X. Wang and D. Zhang, Statistical tongue color distribution and its application. In *Proc. International Conference on Computer and Computational Intelligence* (2011).
- [29] H.Z. Zhang and K.Q. Wang, eds., SVR based color calibration for tongue image. In *Proc. International Conference on Machine Learning and Cybernetics*, 5065–5070 (2005).
- [30] R. Duda and P. Hart, eds., *Pattern Classification, 2nd Edition*. Wiley (2000).
- [31] C. Cortes and V. Vapnik, Support-Vector Networks, *Machine Learning*. **20**, 273–297 (1995).

Chapter 21

Moving Object Detection using Multi-layer Markov Random Field Model

By:
Badri Narayan Subudhi[†], Susmita Ghosh[†] and Ashish Ghosh^{*}

[†]*Department of Electronics and Communication Engineering
National Institute of Technology Goa, India*
subudhi.badri@gmail.com

[†]*Department of Computer Science and Engineering
Jadavpur University, Kolkata, India*
susmitaghoshju@gmail.com

^{*}*Machine Intelligence Unit
Indian Statistical Institute, Kolkata, India*
ash@isical.ac.in

In this work, a multi-layer Markov model based spatio-temporal segmentation scheme for moving object detection is proposed. Spatial segmentation for initial frame is obtained by multi-layer compound Markov random field (MLCMRF) followed by MAP estimation with a combination of simulated annealing (SA) and iterative conditional mode (ICM) techniques. For subsequent frames, a change information based heuristic initialization is proposed for faster convergence of the MAP estimation. For temporal segmentation, label difference based change detection technique is proposed. In one of them, we have considered label frame difference followed by Otsu's thresholding for change detection mask (CDM) generation and in the other, we have proposed label frame difference followed by entropy associated window selection for CDM generation. This CDM is further modified with the video object planes (VOPs) from the previous frame to identify the locations of the moving objects. The results obtained by the proposed technique are compared with those of the existing state-of-the-art techniques and are found to be better.

21.1. Introduction

The first step in video image analysis and object tracking is detection of moving objects [1, 2]. Accuracy of tracking can be enhanced by isolating the background perfectly from the object. Since detection of moving object is highly affected by random fluctuations in the object surface, orientation, texture, scene geometry, noise, detection of moving objects from real life video sequences is always a crucial problem [3]. For a given video, this task can be performed in two ways: (i) temporal segmentation and (ii) spatio-temporal segmentation.

In temporal segmentation, similarity between pixels/ regions of two image frames is considered to isolate the locations of moving objects from their background. Temporal segmentation for moving object detection can be performed in two ways: (i) motion (change) detection and (ii) motion estimation. Change or motion detection is the process of identifying changed and unchanged regions from the extracted video image frames, when the camera is fixed (stationary) and the objects are moving. In motion estimation, it is required to calculate the motion vectors for the target frame with respect to candidate frame to estimate the positions of the moving objects at different instants of time. In case of motion estimation, the camera can be fixed or moving [4].

In spatio-temporal object detection scheme, the objects in the scene are subsequently formed by associating the already formed spatial regions using their low-level features. The object in the initial/ candidate frame of the scene is assumed to be composed of a few regions. A robust approach can be used to get motion information, i.e., to estimate the position of the object in a target frame, based on its current position (i.e., the location in the candidate frame) and its estimated motion features. For getting motion information of the objects in the frames of a video, one can use different temporal segmentation schemes like motion detection (to analyze fixed camera captured sequences) or motion estimation scheme (to analyze fixed and moving camera captured sequences) [5].

A novel spatio-temporal video segmentation scheme is addressed by Fan *et al.* in [6], where the spatial segmentation of the image frame is obtained by entropy based thresholding technique which depends on variance among frames. For temporal segmentation, block-wise motion correlation is considered. However, thresholding based segmentation scheme depends on the gray level information of a scene. Hence, the segmentation result obtained

from this scheme provides disconnected segmented regions rather than a connected one.

It is found from the literature [5] that region growing based segmentation scheme takes care of spatial correlation of pixels in an image frame. Deng and Manjunath [7] proposed an unsupervised method of moving object detection, where spatial segmentation is performed by color quantization followed by region growing. Selection of seed point and boundary preservation is accomplished by considering multi-scale joint image generation (JSEG) scheme. For temporal segmentation, the regions corresponding to objects are matched in temporal direction by computing the motion vectors of object regions in the target frame. To enhance the accuracy of object detection, Kim and Park [8] proposed a new edge based region growing scheme. Here, selection of seed point is accomplished by thresholding the newly proposed edge image. In temporal segmentation, variance of each region (obtained by spatial segmentation scheme) is matched in the temporal direction to detect moving objects.

Watershed algorithm is another approach of performing region based spatial segmentation. An early work using watershed segmentation scheme for object detection is proposed by Wang [9], where the said algorithm is used for object boundary identification. Here, motion estimation along with morphological operation is used for temporal segmentation. Watershed algorithm sometimes provides an over-segmented output, and hence produces an effect of silhouette in object detection [10]. To detect the object boundary properly, Kim and Kim [11] proposed a wavelet transform based object detection scheme. Here they have considered a multi-resolution wavelet transform to get lower resolution of an image frame. Recently, He *et al.* [12] have considered a marker controlled watershed scheme for object detection. They have considered boundary curvature ratio, region homogeneity and boundary smoothness as the criteria for region merging. It is to be noted that, region growing and watershed segmentation suffer from the problem of over-segmentation. Again, in illumination variation condition, these approaches hardly give any good impression.

Generally, variation in gray level intensity, color and appearance of a real world video arises due to non-uniform lighting, random fluctuations in the object surface orientation, textures, scene geometry or noise [13]. These spatial gray level ambiguities of the image frames in a video make the non-statistical spatial segmentation schemes very difficult for moving object detection. Hence, it requires some kind of stochastic methods to model the important attributes of an image frame so that better segmen-

tation result can be obtained than that obtained using non-statistical approach based segmentation scheme. MRF, a well known statistical model, provides a convenient way to model contextual features of an image frame such as pixel gray values, edge, entropy, color, texture, and motion. This is achieved through characterizing mutual influences among such entities using conditional probability distributions. MRF theory provides a way to model the a priori probability of context dependent patterns, such as textures and object features. MAP is one of the most popular statistical criteria for optimality and in fact it has become the most popular choice in vision group. MRFs and the MAP criterion together give rise to the MAP-MRF framework [14].

An early work on MRF based object detection scheme was proposed by Hinds and Pappas [15]. In order to obtain a smooth transition of segmentation results from frame to frame, temporal constraints are introduced. The authors have adhered to a multi-resolution approach to reduce the computational burden. A similar approach, where MRF model has been used to obtain a 3-D spatio-temporal segmentation, is proposed by Wu and Reed [16], where region growing approach along with contour relaxation is applied to obtain accurate boundaries of objects. In the approach proposed by Babacan and Pappas [17], the video sequences are modeled as MRF, and a spatially constrained MRF model is used for spatial segmentation. Segmentation results of previous frame acted as the precursor for segmentation of the next frame.

Most of the spatio-temporal segmentation schemes proposed in the literature were affected by the effect of silhouette. The effect of silhouette is found to be less in a recently proposed moving object detection technique of Kim and Park [18], where the MAP of the MRF model is estimated by distributed genetic algorithms. Thereafter for temporal segmentation of the current frame, the CDM is updated with the label information of the current and the previous frames. Combination of both spatio-temporal spatial segmentation and temporal segmentation is performed to obtain the video object plane (VOP).

A region labeling approach that uses MRF model with motion estimation is explored by Tsaig and Averbuch [19] to obtain the VOPs. Here, to obtain an initial partition of the considered frame, watershed algorithm is used. Recently, a region based object detection algorithm is proposed by Huang *et al.* [20] to obtain a set of motion coherent regions. The authors have also used MRFs for spatial segmentation and integrated the spatial as well as temporal sequences to obtain the moving objects. Jodoin *et al.* [21]

have proposed a robust moving object detection and tracking scheme for both fixed and moving camera captured video sequences where MRF is used for fusion of label fields.

It is evident from the literature that, spatio-temporal segmentation techniques rely on the accuracy of spatial segmentation. Generally, a video scene contains illumination variation, noise and motion blurring. Use of conventional segmentation techniques is likely to be affected by these and hence will give rise to inaccurate segmentation maps. Inaccurate spatial segmentation may cause effects of silhouette in object detection and tracking [22].

In this chapter, we propose a multi-layer compound Markov model based spatio-temporal object detection technique that can detect moving objects with less effects of silhouette. The proposed scheme is a combination of both spatio-temporal spatial segmentation and temporal segmentation techniques. Here, at first, we obtain spatio-temporal spatial segmentation for a given initial frame by multi-layer compound Markov random field (MLCMRF). The proposed MLCMRF uses five Markov models in a single framework, one in the spatial direction using color features, and four in temporal directions (using two color features and two edge map/line fields). Hence, the proposed MLCMRF uses spatial distribution of color, temporal color coherence and edge maps in the temporal frames. Thereafter, for subsequent frames, segmentation is obtained by adding some change information of these frames with the initial frame segmentation result. The spatio-temporal spatial segmentation problem is formulated using MAP estimation principle. For the initial image frame, for fast convergence, the MAP estimate is obtained using a combination of simulated annealing (SA) and iterative conditional mode (ICM) procedures in a sequential manner. For subsequent frames, to obtain a rough initialization, original pixels corresponding to the changed regions (changes obtained between the current and the previous frames) of the current frame are super-imposed on previously available segmented frame. Subsequent frames are modeled with MLCMRF and the MAP estimate is obtained by ICM algorithm starting from this initialization. This spatio-temporal spatial segmentation combined with temporal segmentation yields the VOP and hence can detect moving objects.

For temporal segmentation, we have proposed two label difference based change detection techniques. For this, we obtain the difference image by considering the label information of the pixels from the spatially segmented output of two image frames. For thresholding the difference image, we have

proposed two techniques. In one approach, we have considered label difference image with Otsu's thresholding for CDM generation and in the second one, we propose a local histogram thresholding scheme to segment the difference image by dividing it into a number of small non-overlapping regions/windows and thresholding each window separately. The window/block size is determined by measuring the entropy content of it. The segmented regions from each of the window are concatenated to find the (entire) segmented image. The thresholded difference image is called the CDM and it represents the changed regions corresponding to the moving objects in the given image frame.

The proposed scheme is tested on several video sequences and three of them are reported in this chapter. The results obtained by the proposed spatio-temporal spatial segmentation method are compared with those of edge less [22], edge based [22], JSEG [7] and meanshift [23] methods of segmentation and are found to be better. Computational time requirement for the proposed method is less compared to edge less and edge based approaches. Similarly, the VOPs obtained by the proposed technique are compared with those of CDM with a gray level difference and Otsu's thresholding, and it is found that the VOPs produced by the proposed techniques are better.

Organization of this chapter is as follows. Section 21.2 contains the spatial segmentation method using spatio-temporal framework. Temporal segmentation based on the proposed CDMs is elaborated in Section 21.3. Section 21.4 provides simulation results and analysis. Discussion and conclusion is presented in Section 21.5.

21.2. Spatial Segmentation using Multi-layer Compound Markov Random Field Model

In the spatio-temporal spatial segmentation scheme, we have modeled each video image frame with MLCMRF and the segmentation problem is solved using the MAP estimation principle. For initial frame segmentation, a combination of SA and ICM techniques is used to obtain the MAP estimate. For segmentation of other frames, the change information between the frames are incorporated with the previously available segmented frame so as to have an initialization.

21.2.1. Framework for Maximum A Posterior Probability (MAP) Estimation for MRF

Let the observed video sequence y be a 3-D volume consisting of spatio-temporal image frames. y_t represents a video image frame at time t and hence is a spatial entity. Each pixel in y_t is a site s denoted by y_{st} . Let Y_t represent a random field and y_t be a realization of it. Thus, y_{st} denotes a spatio-temporal co-ordinate of the grid (s, t) . Let x denote the segmentation of video sequence y and x_t denote the segmented version of y_t . Similarly, x_{st} represents label of a site in the t^{th} frame. Let us assume that X_t represents the MRF from which x_t is a realization.

For any segmentation problem, the label field can be estimated by maximizing the following posterior probability distribution.

$$\hat{x}_t = \arg \max_{x_t} P(X_t = x_t | Y_t = y_t). \quad (21.1)$$

Using Bayes' framework we can derive

$$\hat{x}_t = \arg \max_{x_t} \frac{P(Y_t = y_t | X_t = x_t)P(X_t = x_t)}{P(Y_t = y_t)}, \quad (21.2)$$

where \hat{x}_t denotes the estimated labels. The prior probability $P(Y_t = y_t)$ is constant and hence Eq. (21.2) reduces to

$$\hat{x}_t = \arg \max_{x_t} P(Y_t = y_t | X_t = x_t, \theta)P(X_t = x_t, \theta); \quad (21.3)$$

where θ is the parameter vector associated with the clique potential function of x_t . Here, \hat{x}_t is the MAP estimate. Equation (21.3) contains two parts: $P(X_t = x_t)$, the prior probability and $P(Y_t = y_t | X_t = x_t, \theta)$ as the likelihood function.

The prior probability $P(X_t = x_t)$ can be expressed as

$$P(X_t = x_t, \theta) = \frac{1}{z} e^{\frac{-U(x_t)}{T}} = \frac{1}{z} e^{\frac{-\{\sum_{c \in C} V_c(x_t)\}}{T}}, \quad (21.4)$$

where z is the partition function expressed as $z = \sum_{x_t} e^{\frac{-U(x_t)}{T}}$, $U(X_t)$ is the energy function (a function of clique potentials). In Eq. (21.4), $V_c(x_t)$ represents the clique potential function in spatial domain. It can be defined in terms of MRF model bonding parameter as

$$V_c(x_t) = \begin{cases} +\alpha & \text{if } x_{st} = x_{qt} \\ -\alpha & \text{if } x_{st} \neq x_{qt}, \end{cases} \quad (21.5)$$

where α is the MRF model bonding parameter and q is a neighboring site of s .

21.2.2. Multi-layer Compound Markov Random Field Model

In the MLCMRF, an MRF model is considered in spatial direction. Similarly, pixels in the temporal direction are also modeled as MRFs. We have considered the second order MRF modeling both in spatial as well as in temporal directions. In order to preserve the edge features, another MRF model is considered with the line fields of x_t and those of x_{t-1} and x_{t-2} . It is known that if X_t is an MRF then it satisfies the Markovianity property [24] in spatial direction; i.e.,

$$P(X_{st} = x_{st} \mid X_{qt} = x_{qt}, \forall q \in S, s \neq q) = P(X_{st} = x_{st} \mid X_{qt} = x_{qt}, (q, t) \in \eta_{st});$$

where η_{st} denotes the neighborhood of (s, t) and S denotes the spatial lattice of X_t . For temporal MRF, the following Markovianity property is also satisfied:

$$\begin{aligned} p(X_{st} = x_{st} \mid X_{qr} = x_{qr}, s \neq q, t \neq r, \forall (s, t), (q, r) \in V) \\ = p(X_{st} = x_{st} \mid X_{qr} = x_{qr}, s \neq q, t \neq r, (q, r) \in \eta_{sr}). \end{aligned}$$

Here, V denotes the 3-D volume of the video sequence and η_{sr} denotes the neighborhood of (s, t) in temporal direction. We have considered s as a spatial site, q as a temporal site and e as an edge site. The set of all the sites is represented by S . Therefore, $S = \{s\} \cup \{q\} \cup \{e\}$. In spatial domain, X_t represents the MRF model of x_t and hence the prior probability can be expressed as Gibb's distribution with $P(X_t) = \frac{1}{z} e^{-\frac{U(X_t)}{T}}$. We have considered the following clique potential functions for the present work.

$$V_{sc}(x_{st}, x_{qt}) = \begin{cases} +\alpha & \text{if } x_{st} \neq x_{qt} \text{ and } (s, t), (q, t) \in S \\ -\alpha & \text{if } x_{st} = x_{qt} \text{ and } (s, t), (q, t) \in S. \end{cases}$$

Analogously, in temporal direction

$$V_{tec}(x_{st}, x_{qr}) = \begin{cases} +\beta & \text{if } x_{st} \neq x_{qr}, (s, t), (q, r) \in S, t \neq r, \text{ and } r \in \{(t-1), (t-2)\} \\ -\beta & \text{if } x_{st} = x_{qr}, (s, t), (q, r) \in S, t \neq r, \text{ and } r \in \{(t-1), (t-2)\}; \end{cases}$$

and for the edge map in the temporal direction

$$V_{teec}(x_{st}, x_{er}) = \begin{cases} +\gamma & \text{if } x_{st} \neq x_{er}, (s, t), (e, r) \in S, t \neq r, \text{ and } r \in \{(t-1), (t-2)\} \\ -\gamma & \text{if } x_{st} = x_{er}, (s, t), (e, r) \in S, t \neq r, \text{ and } r \in \{(t-1), (t-2)\}. \end{cases}$$

$V_{sc}(x_{st}, x_{qt})$, $V_{tec}(x_{st}, x_{qr})$ and $V_{teec}(x_{st}, x_{er})$ denote spatial, temporal and edge clique potential functions, respectively. Here, α , β , and γ are the parameters associated with the clique potential functions. These are positive constants and are determined on trial and error basis. In an image modeling, the clique potential function is the combination of the above three terms. Hence, the energy function is of the following form:

$$U(X_t) = \sum_{c \in C} V_{sc}(x_{st}, x_{qt}) + \sum_{c \in C} V_{tec}(x_{st}, x_{qr}) + \sum_{c \in C} V_{teec}(x_{st}, x_{er}). \quad (21.6)$$

Figure 21.1 shows the diagrammatic representation of the MLCMRF modeling. Figure 21.1(a) depicts that each site s at location (a, b) is an MRF modeled with its neighbors in spatial direction while Fig. 21.1(b) denotes the diagram for another MRF model in temporal direction. Each site s at location (a, b) in the t^{th} frame is modeled with neighbors of the corresponding pixels in temporal direction, i.e., in $(t - 1)^{th}$ and $(t - 2)^{th}$ frames. Similarly, an MRF model that takes care of edge features is considered by modeling the line field of the t^{th} frame with the neighbors of the corresponding pixels in the $(t - 1)^{th}$ and $(t - 2)^{th}$ frames. The MRF model diagram for line field is shown in Fig. 21.1(c).

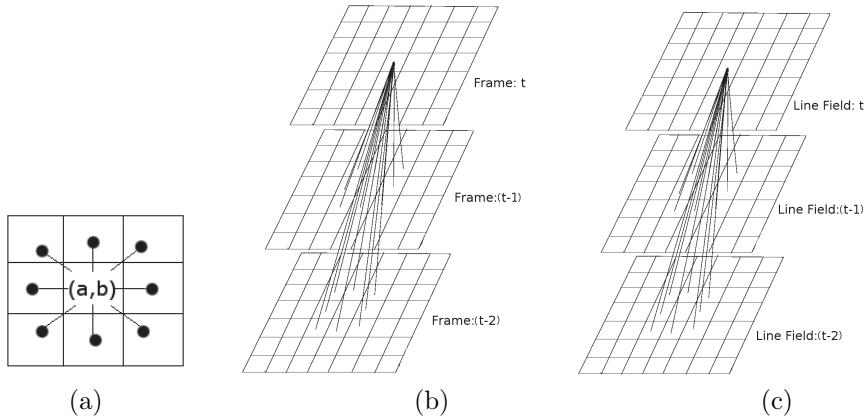


Fig. 21.1. (a) Neighborhood of a site for MRF modeling in the spatial direction, (b) MRF modeling taking two previous frames in the temporal direction, and (c) MRF modeling using two additional frames with line fields to take care of edge features.

21.2.3. MAP Estimation-based Segmentation Scheme for Initial Frame

The observed image sequence y is assumed to be a degraded version of the actual image sequence x . The degradation process is assumed to be Gaussian. Thus, the label field x_t can be estimated from the observed random field Y_t by maximizing the posterior probability distribution as given in Eq. (21.3). Considering the energy function as in Eq. (21.6), the prior probability $P(X_t = x_t)$ (using the three clique potentials as given in

the earlier section) is described as

$$\begin{aligned} P(X_t = x_t) &= \frac{1}{z} e^{-\frac{U(x_t)}{T}} \\ &= \frac{1}{z} e^{-\frac{1}{T} \sum_{c \in C} \{V_{sc}(x_{st}, x_{qt}) + V_{tec}(x_{st}, x_{qr}) + V_{teec}(x_{st}, x_{er})\}}. \end{aligned} \quad (21.7)$$

Considering the pixel features to be independent, we can have a simplified form of the likelihood function. $P(Y_t = y_t | X_t = x_t, \theta)$ can be expressed as

$$\begin{aligned} P(Y_t = y_t | X_t = x_t, \theta) \\ = \prod_{s \in S} \frac{1}{\sqrt{(2\pi)^f \det[\Sigma_{Q_t}]}} e^{-\frac{1}{2} (y_{st} - \mu_{Q_t})^T \Sigma_{Q_t}^{-1} (y_{st} - \mu_{Q_t})}, \end{aligned} \quad (21.8)$$

where Σ_{Q_t} is the covariance matrix, $\det[\Sigma_{Q_t}]$ represents the determinant of matrix Σ_{Q_t} and f is the number of features (for color image, the three features R, G, and B are used and it is assumed that decorrelation exists among the three RGB planes). Each pixel (i.e., site s) class in frame t denoted by $Q_t = \{q_1, q_2, \dots, q_K\}$ is represented by its mean vector μ_{Q_t} and the covariance matrix Σ_{Q_t} .

Therefore, the MAP estimate can be written as:

$$\hat{x}_t = \arg \max_{x_t} \left\{ \sum_{s \in S} \left\{ A - \left[\frac{1}{2} (y_{st} - \mu_{Q_t})^T \Sigma_{Q_t}^{-1} (y_{st} - \mu_{Q_t}) \right] \right\} - \left[\sum_{c \in C} \{V_{sc}(x_{st}, x_{qt}) + V_{tec}(x_{st}, x_{qr}) + V_{teec}(x_{st}, x_{er})\} \right] \right\}, \quad (21.9)$$

where $A = -\frac{1}{2} \log((2\pi)^f \det[\Sigma_{Q_t}])$ and \hat{x}_t , the MAP estimate, is obtained by a combination of SA and ICM algorithms.

21.2.4. Change Information-based Segmentation Scheme for Subsequent Frames

It is to be noted that spatio-temporal spatial segmentation in MRF-MAP framework is computation intensive when the algorithm is initialized randomly. To reduce computational time, in this work, we propose a change information based heuristic initialization technique. To initialize the processing of the current frame, in addition to the previously segmented frame, the change information between the present as well as the previously considered frames is considered. The change information is obtained by computing the absolute values of the intensity difference between the current and the previously considered frames followed by a thresholding approach.

The pixel values corresponding to the changed region of the current frame are superimposed on the previously available segmented frame for initialization.

Let $y_{(t-d)}$ denote a frame at time $(t-d)$ whose spatio-temporal spatial segmentation $x_{(t-d)}$ is available with us. Now considering y_t as a frame at an instant t , $x_{(t)i}$ represents its initialization obtained by change information. x_t represents its final spatio-temporal spatial segmentation. Initialization for segmenting $y_{(t)}$ can be done as follows:

- i. Obtain the changed region corresponding to the frame y_t by taking a difference of the gray values of the frames y_t and $y_{(t-d)}$ followed by thresholding, and the changes thus produced is denoted by $y_{t|y_t-y_{(t-d)}|}$.
- ii. The pixels corresponding to these changed regions in the segmentation result of the $(t-d)^{th}$ frame, $x_{(t-d)}$, are initialized as

$$x_{t(t-d)} = x_{(t-d)} - x_{(t-d)|y_t-y_{(t-d)}|}. \quad (21.10)$$

To initialize the t^{th} frame, the regions in $x_{t(t-d)}$ are replaced by the original gray values of y_t . This is done as follows.

$$x_{(t)i} = x_{t(t-d)} + y_{t|y_t-y_{(t-d)}|}, \quad (21.11)$$

where $y_{t|y_t-y_{(t-d)}|}$ represents the changes which took place between the t^{th} and the $(t-d)^{th}$ frames. $x_{(t)i}$ serves as the initialization for spatio-temporal spatial segmentation of the t^{th} frame in MLCMRF model. It is to be noted that to obtain x_t , ICM is executed on the t^{th} frame starting from $x_{(t)i}$.

21.3. Temporal Segmentation and VOP Generation

Generally, temporal segmentation is performed to classify the foreground and the background in a video image frame. In temporal segmentation, a CDM is obtained which serves as a precursor for detection of the foreground as well as the background. The general procedure for obtaining the CDM is to take a difference between the gray value of the current and that of the previously considered frames followed by a thresholding algorithm. In this work, we have proposed two different ways for CDM generation: global histogram thresholding and entropy based window selection for local thresholding. Both of these approaches follow three steps: construction of difference image, CDM generation, and modification of CDM for moving object detection. In order to detect the moving objects in absence of any reference frame, some information from the previous frame is used to update the current CDM.

21.3.1. Generation of Difference Image

As opposed to the CDM obtained using conventional gray value difference, we have considered a label frame difference CDM, where silhouette effect is expected to be less. Let us consider two image frames at the t^{th} and the $(t - d)^{th}$ instants of time. For temporal segmentation, we have obtained a difference image by considering the label information of the pixels from the spatial segmentation of the two image frames. For difference image generation, at a particular pixel location, if the results of the spatial segmentation of the two image frames are found to be similar, then the difference image value is set to 0; otherwise the pixel value in the difference image is obtained by taking a difference of the corresponding values of the R , G and B components of the considered frames. The use of such a difference image has an advantage of preserving the boundary information of the object. Usually, the boundary of the object has a higher chance of misclassification. By considering a difference image with label information, the amount of misclassification can be reduced. It is also found that the isolated pixels in the background regions may have large variation in gray value due to noise and illumination variations. Labeled information in difference image can also reduce such effects.

21.3.2. Generation of Change Detection Mask (CDM)

To obtain the changes between consecutive image frames, thresholding operation is carried out on the difference image [25]. In the present work, two thresholding techniques are proposed: global thresholding and entropy based window selection for local thresholding. These are discussed below.

21.3.2.1. Global Thresholding

For temporal segmentation, we have obtained the difference image with labeled information by using the difference of the respective R , G and B components of the considered frames and their spatial segmentation information. We applied Otsu's thresholding algorithm [26] to each channel (i.e., R , G and B) of the difference image. Once the thresholded images for all the channels are obtained, they are fused by a logical *OR* operator.

The CDM, thus obtained, is of two types: changed and unchanged (denoted by 1 and 0) and is represented as

$$CDM(a, b) = \begin{cases} 1; & \text{if it is changed} \\ 0; & \text{otherwise.} \end{cases}$$

21.3.2.2. Entropy-based Window Selection for Local Thresholding

Determination of the threshold value for each channel of the labeled difference image is a very difficult task. The following two situations may occur when a global thresholding algorithm is used on the difference image: (i) a few pixels which actually correspond to the background in an image frame are identified as changed pixels, and (ii) a few pixels in the difference image which are originally from the changed regions and lie on the lower range of the histogram are identified as unchanged ones. An adaptive thresholding approach, where the image is divided into small regions/ windows and then thresholding is done on each small region, is used to overcome this limitation. However, the choice of such small regions/ windows is a difficult task. If a large window is considered, smaller changes can not be detected properly. On the contrary, in a small sized window noise or pixels affected by illumination variation are more and the corresponding pixels may be detected as object ones. In this regard, we propose an entropy based adaptive window selection scheme to determine the size of the blocks/ windows. The threshold value for a window is obtained by Otsu's scheme [26]. A union of all the thresholded blocks represents the CDM. The methodology is described below in detail.

The basic notion of window selection approach is to fix the window size primarily focussing on the information content of the window or the sub-image. In other words, fixing the size of a window depends on the entropy of the said window. In this approach, initially an arbitrarily small window (in the present case, 5×5) is considered (at the beginning of the image) and the entropy of the window is computed from its gray level distribution and is denoted as

$$H_w = \sum_{i=1}^{N_g} p_i \log_e \left(\frac{1}{p_i} \right), \quad (21.12)$$

where p_i is the probability of occurrence of the i^{th} gray level in the window and N_g is the maximum gray level. It is needless to mention that the local entropy is related to the variance of gray value of the window or the sub-image. Entropy is more for a heterogeneous region, and less for a homogeneous region. Hence, object-background transition regions will have more local entropy than that in non-transition regions of the image. It is observed that the information content of a heterogenous region in an image is close to some fraction of the entropy of the whole image. If the entropy of a window is greater than some fraction of the entropy of the

whole image (i.e., $H_w > Th$, where $Th = c * H_{di}$; Th is the threshold, c is a constant in $[0, 1]$, and H_{di} is the entropy of the difference image di), then the said window is chosen for segmentation. Otherwise, the size of the window is incremented by Δw (here, it is considered as 2) and the condition is tested again. Once a window has been selected for segmentation, the next window is selected from the remaining portions of the difference image. The final thresholded image is obtained by taking a union of all the considered thresholded small (images) windows.

21.3.3. Modification of Change Detection Mask and VOP Generation

The CDM obtained using global thresholding or entropy based window selection for thresholding contains either changed or unchanged (denoted as, 1 or 0) pixels. However, in the generated CDM, the previous position of the moving object (excluding the overlapping area) is labeled as a changed area, and the same is required to be eliminated in the current frame. To detect the moving object in the current frame, modification of the CDM is done in a way so that the object location in the previous frame is eliminated and the overlapped area corresponding to the moving object in the subsequent frame is highlighted. To improve this result further, output of the change detection, thus obtained, is combined with the VOP of the previous frame based on label information of the current and the previous frames. The VOP of the previous frame, the $(t - d)^{th}$ one, is represented as a matrix (of size $M \times N$) as

$$R(t - d) = \{r_{a,b}(t - d) | 0 \leq a \leq (M - 1), 0 \leq b \leq (N - 1)\},$$

where each element of the matrix, i.e., $r_{a,b}(t - d)$ represents the value of the VOP at location (a, b) in the $(t - d)^{th}$ frame. Here, $R(t - d)$ is a matrix having the same size as that of the image frame (i.e., $M \times N$) and the $(a, b)^{th}$ location represents the a^{th} row and the b^{th} column. Thus

$$r_{a,b}(t - d) = \begin{cases} 1; & \text{if it is in the object,} \\ 0; & \text{if it is in the background.} \end{cases}$$

After obtaining a temporal segmentation of a frame at time t , we get a binary output with object as one class (denoted as, FM_t) and the background as other (denoted as, BM_t). The region, forming the foreground part in the temporal segmentation is identified as moving object region, and the pixels corresponding to the FM_t part of the original frame y_t form the VOP.

21.4. Results and Discussion

As mentioned earlier, three datasets are used to assess the effectiveness of the proposed method. The considered video sequences are *Karlsruhe-I*, *Karlsruhe-II* and *Claire*. To have a quantitative evaluation of the proposed scheme, two ground-truth based performance measures are considered: one for the proposed spatial segmentation scheme and the other for the resultant moving object locations. For evaluating the accuracy of spatial segmentation, pixel by pixel comparison of the ground-truth image is made with the produced spatial segmentation results (also called the number of misclassified pixels). To evaluate the performance of moving object detection, precision, recall and f-measures are taken into consideration [27]. To validate the proposed scheme, average of these measures (by taking the mean value over all the image frames of the considered video sequences) are considered. Hence, the measures are named as average number of misclassified pixels, average precision, average recall and average f-measure. It may be noted that higher the values of average precision, average recall and average f-measure, better is the result. On the other hand, the average number of misclassified pixels needs to be minimized.

Karlsruhe-I sequence contains three moving objects and each one is moving with a different speed. This video contains moving objects like: a black car, a white car and a person (moving in lawn). Figure 21.2(a) shows a few image frames of this sequence (considered for experimentation). Corresponding manually constructed ground-truth images are shown in Fig. 21.2(b). The spatial segmentation results obtained by the proposed scheme are shown in Fig. 21.2(c). The spatial segmentation results of these frames using edge less scheme are displayed in Fig. 21.2(d). It is observed from these results that a few portions of the moving black car and the person in the lawn are merged into background. It is also observed that a few still objects are also merged with the background. The JSEG based spatial segmentation results of these frames are displayed in Fig. 21.2(e). These results show that the person and the black car got merged into the background. Similarly, from the results obtained by meanshift scheme (Fig. 21.2(f)), it is seen that the black car and the person moving in the lawn are merged into background. Some portions of the white car are also merged into background.

It is found from Figs. 21.2(g) and 21.2(h) that the gray level CDM with global thresholding is not able to properly detect the objects, particularly the black car and the man. Similarly, the rear end of the white car is also not

detected properly thereby resulting in object and background misclassification. Similar results are also obtained by label frame based CDM approach as shown in Figs. 21.2(i) and 21.2(j). As observed from Fig. 21.2(k) and Fig. 21.2(l), the objects (black car and the man) along with the white car is detected properly using adaptive thresholding.

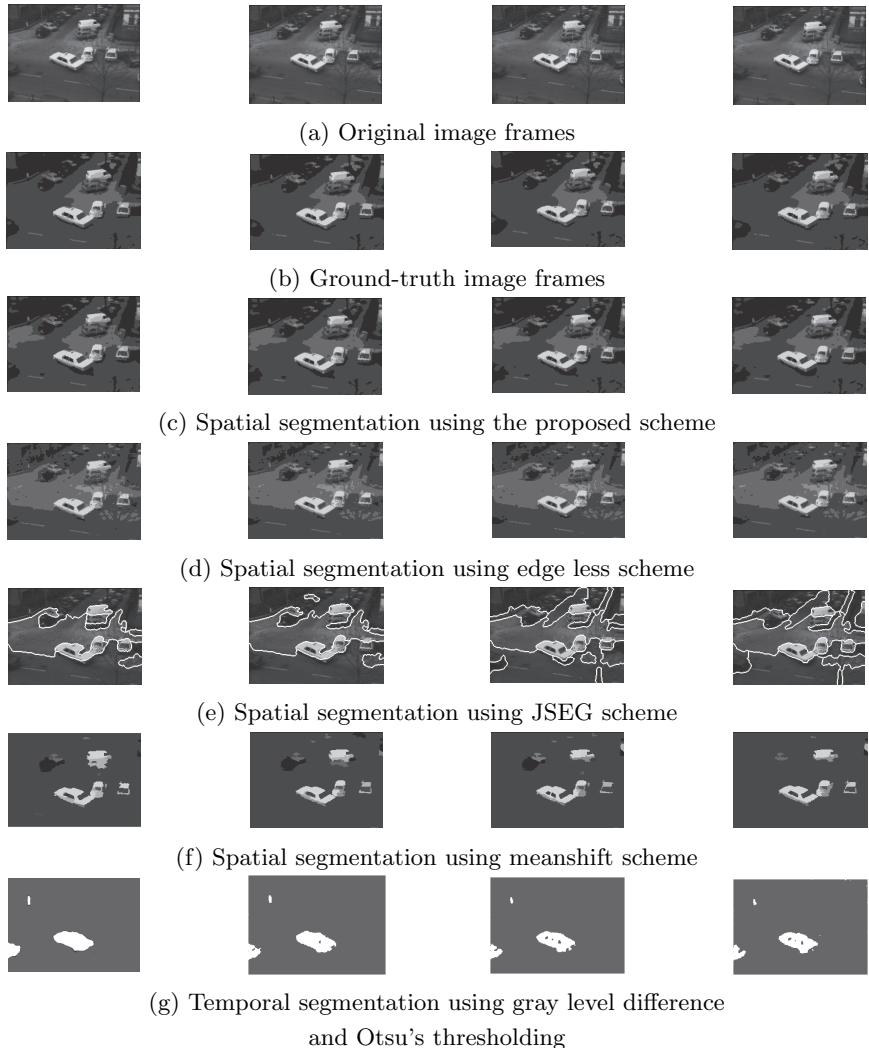


Fig. 21.2. Results for spatial & temporal segmentations and VOP generation for Karlsruhe-I video sequence using different techniques.

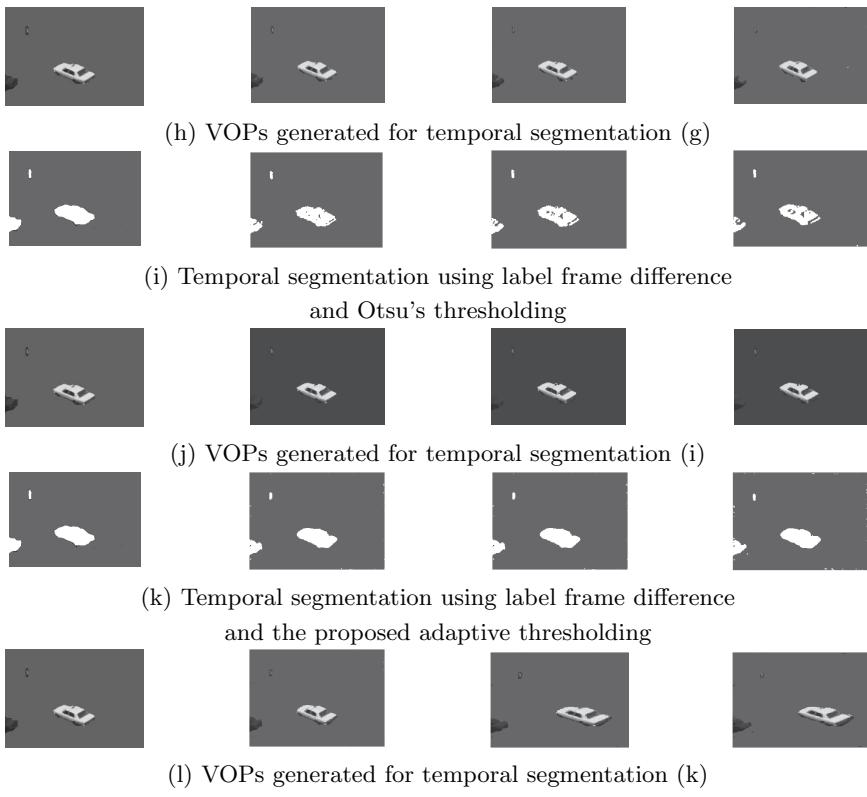


Fig. 21.2 (Continued)

Figure 21.3(a) shows a few image frames of *Karlsruhe-II* sequence (having two moving objects). This is an illumination variate noisy sequence. Figure 21.3(b) provides the corresponding ground-truth image. The proposed change information based spatial segmentation results of these frames are shown in Fig. 21.3(c). Corresponding results using edge less, JSEG and meanshift approaches of segmentations are shown in Figs. 21.3(d)–21.3(f). Otsu's global thresholding with gray level difference CDM produces results where many parts of moving objects are missing (shown in Fig. 21.3(g) and Fig. 21.3(h)). Figure 21.3(i) and Fig. 21.3(j) show the moving object detection results using label frame difference CDM with Otsu's global thresholding technique. It is observed that gray level difference followed by Otsu's thresholding and label frame difference followed by Otsu's thresholding were not able to detect the moving objects in the scene properly and give rise an effect of silhouette in the scene. Figure 21.3(k) shows

the temporal segmentation results obtained using the proposed adaptive window selection scheme. It is seen from the results that all the parts of the moving objects have been detected with less (object background) misclassification (shown in Fig. 21.3(l)). It is found that using Otsu's global thresholding scheme, one of the moving cars is almost missed whereas the proposed adaptive window based scheme provided better results.

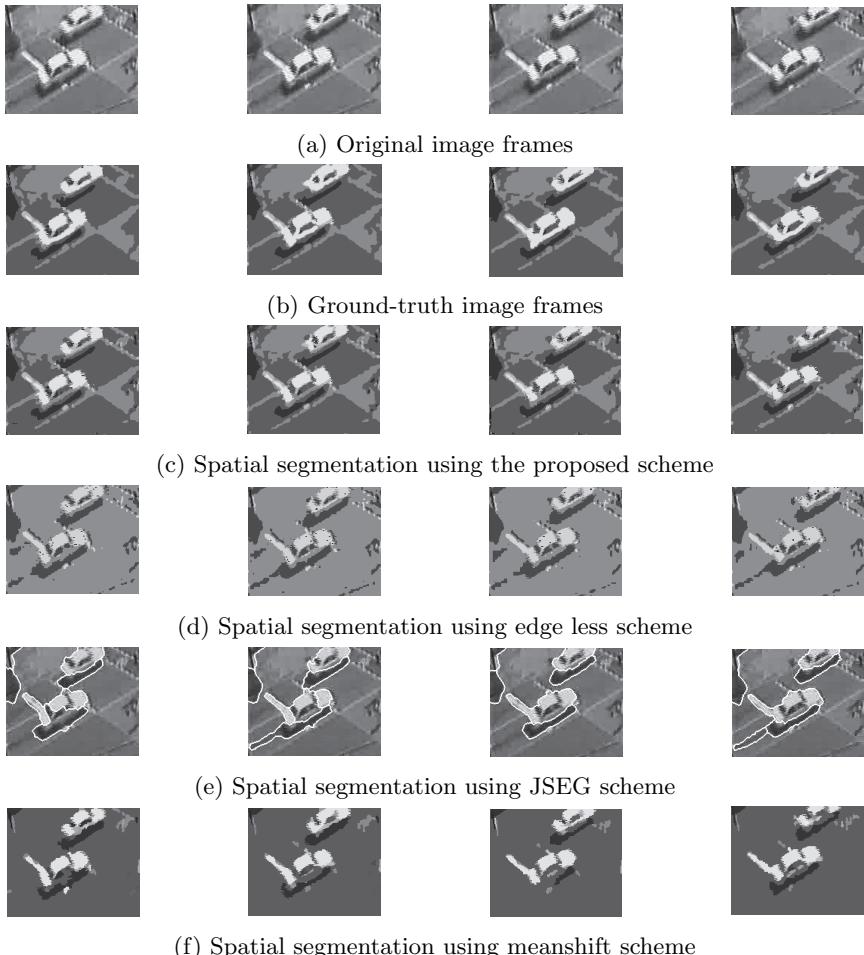


Fig. 21.3. Results for spatial & temporal segmentations and VOP generation for Karlsruhe-II video sequence using different techniques.



(g) Temporal segmentation using gray level difference
and Otsu's thresholding



(h) VOPs generated for temporal segmentation (g)



(i) Temporal segmentation using label frame difference
and Otsu's thresholding



(j) VOPs generated for temporal segmentation (i)



(k) Temporal segmentation using label frame difference
and the proposed adaptive thresholding



(l) VOPs generated for temporal segmentation (k)

Fig. 21.3 (Continued)

As mentioned earlier, experiment is carried out on *Claire*, a news reader sequence, (here the moving object is ‘Claire’). Figure 21.4(a) shows a few original image frames and Fig. 21.4(b) shows corresponding ground-truth image of this sequence. The spatial segmentation results of the proposed scheme and those obtained using different state-of-the-art techniques are provided in Figs. 21.4(c)–(f). It may be observed that the proposed scheme

is giving better results as compared to the considered existing techniques. Results of moving object detection and corresponding VOPs obtained by gray level difference and Otsu's thresholding are depicted in Figs. 21.4(g) and 21.4(h). It is observed that in some parts of 'Claire', particularly, in the head regions, effect of silhouette is present thereby introducing object background misclassification. The corresponding results obtained by label frame difference and Otsu's thresholding are shown in Figs. 21.4(i) and 21.4(j). Effect of silhouette is also present here. Moreover, many parts of the head of 'Claire' are missing. However, the results obtained by label frame difference with adaptive window selection technique are found to be better (as shown in Figs. 21.4(k) and 21.4(l)).

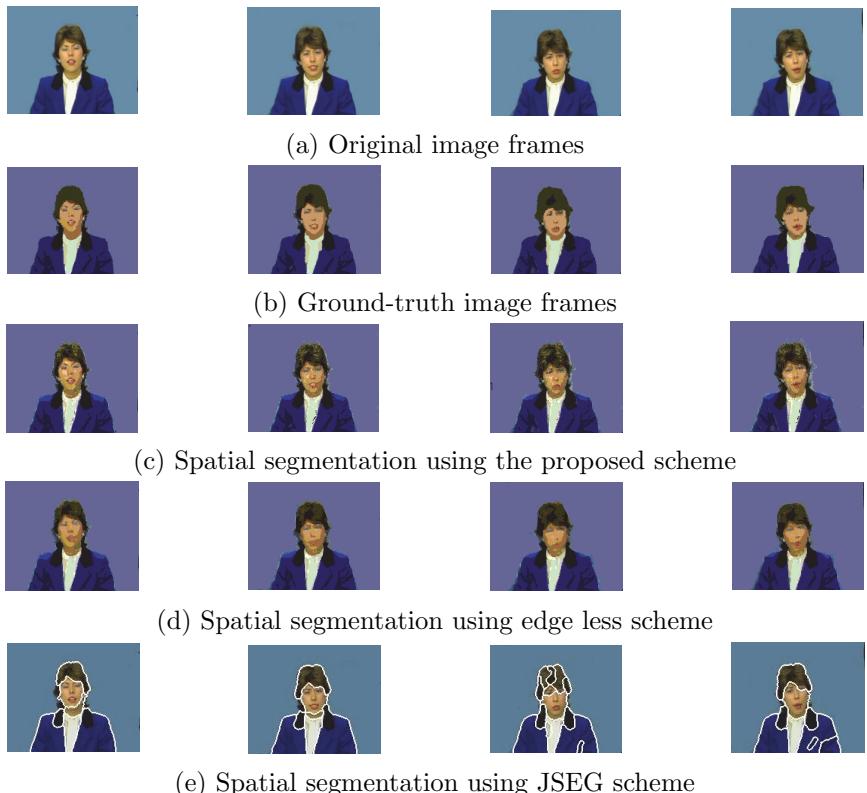


Fig. 21.4. Results for temporal segmentation and VOP generation for Claire video sequence using different techniques.



(f) Spatial segmentation using meanshift scheme

(g) Temporal segmentation using gray level difference
and Otsu's thresholding

(h) VOPs generated for temporal segmentation (b)

(i) Temporal segmentation using label frame difference
and Otsu's thresholding

(j) VOPs generated for temporal segmentation (d)

(k) Temporal segmentation using label frame difference
and the proposed adaptive thresholding

(l) VOPs generated for temporal segmentation (f)

Fig 21.4. (*Continued*)

It is also observed that the average number of misclassified pixels for spatial segmentation is quite less in case of edge based spatial segmentation approach than that of edge less, JSEG and meanshift approaches of segmentation as shown in Table 21.1. The average precision, average recall and average f-measures values for all the considered sequences are put in Table 21.2. The entries in the table reveal that the values of the above three performance measuring indices are more for the proposed scheme than those obtained using the non-window based global thresholding schemes. Hence, we may conclude that the VOPs obtained by the proposed entropy based adaptive (window based) thresholding scheme are better (a less effect of silhouette is noticed) than those obtained by the non-window based global thresholding schemes.

Table 21.1. Average number of misclassified pixels after spatial segmentation.

Video sequences used	Methods used				
	Edge less	Edge based (proposed)	Change information (proposed)	JSEG	Meanshift
<i>Karlsruhe – I</i>	910	518	612	1723	2340
<i>Karlsruhe – II</i>	708	332	518	1115	990
<i>Claire</i>	512	417	492	1418	517

Table 21.2. Average precision, recall, and f-measure values with label frame difference and the proposed adaptive window based thresholding.

Video sequences used	Methods used								
	Algo1 ¹			Algo2 ²			Algo3 ³		
	Pr	Re	F	Pr	Re	F	Pr	Re	F
<i>Karlsruhe – I</i>	0.85	0.80	0.83	0.86	0.85	0.85	0.90	0.94	0.92
<i>Karlsruhe – II</i>	0.86	0.83	0.84	0.89	0.87	0.88	0.93	0.90	0.91
<i>Claire</i>	0.81	0.89	0.85	0.84	0.90	0.87	0.90	0.93	0.91

¹Gray level CDM+Otsu's thresholding,

²Label information based difference+Otsu's thresholding,

³Label information based difference+Proposed thresholding,

Pr: precision, Re: recall and F: f-measure

Experiment is carried out on a *Pentium4(D), 3GHz, L2 cache 4MB, 1GB RAM, 667 FSB* PC with *Fedora – Core* operating system and *C* programming language.

The average time consumed by various techniques to segment a single frame is provided in Table 21.3. By comparing the computational time requirement of all the approaches it can be inferred that the proposed change information based approach is faster than edge less and edge based approaches of segmentation.

Table 21.3. Average time (in second) required for execution of different algorithms.

Video sequences used	Methods used		
	Edge less	Edge based	Change information
<i>Karlsruhe – I</i>	124.7	120.1	11.0
<i>Karlsruhe – II</i>	150.4	125.0	11.0
<i>Claire</i>	100.0	94.0	8.0

21.5. Discussion and Conclusion

In this chapter, a change information based moving object detection scheme is proposed. Spatio-temporal spatial segmentation of the initial frame is obtained by edge based MRF modeling and a combination of SA & ICM for MAP estimation. The segmentation results of the first frame together with some change information from other frames are used as initialization for segmentation of other frames. Then, an ICM algorithm is used on the said frame with such initialization to segment that frame. It is found that the proposed approach produces better segmentation compared to those of edge less, JSEG and meanshift segmentation schemes and comparable results with those obtained using edge based approach. The proposed method gives better accuracy and is found to be approximately 13 times faster compared to the considered MRF based segmentation schemes. The parameters for MRF model are chosen on trial and error basis. For temporal segmentation, two CDMs based on label frame difference followed by Otsu's global thresholding and label frame difference followed by entropy based adaptive local thresholding techniques are used. This reduces the effect of silhouette on the generated VOPs.

References

- [1] A. Yilmaz, O. Javed, and M. Shah, Object tracking : A survey, *ACM Computing Surveys*. **38**(4), 1264–1291 (2006).
- [2] B. Tian, Y. Li, B. Li, and D. Wen, Rear-view vehicle detection and tracking by combining multiple parts for complex urban surveillance, *IEEE Transactions on Intelligent Transportation Systems*. **15**(2), 597–606 (2014).
- [3] K. Lin, S.-C. Chen, C.-S. Chen, D.-T. Lin, and Y.-P. Hung, Abandoned object detection via temporal consistency modeling and back-tracing verification for visual surveillance, *IEEE Transactions on Information Forensics and Security*. **10**(7), 1359–1370 (2015).
- [4] B. N. Subudhi, S. Ghosh, and A. Ghosh, Application of Gibbs-Markov random field and Hopfield-type neural networks for detecting moving objects from video sequences captured by static camera, *Soft Computing*. pp. 1–13 (2014). doi: 10.1007/s00500-014-1440-4.

- [5] P. Salembier and F. Marqués, Region based representation of image and video segmentation tools for multimedia services, *IEEE Transactions on Circuits and Systems for Video Technology*. **9**(8), 1147–1169 (1999).
- [6] J. Fan, J. Yu, G. Fujita, T. Onoye, L. Wu, and I. Shirakawa, Spatiotemporal segmentation for compact video representation, *Signal Processing: Image Communication*. **16**(6), 553–566 (2001).
- [7] Y. Deng and B. S. Manjunath, Unsupervised segmentation of color-texture regions in images and video, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **23**(8), 800–810 (2001).
- [8] B. G. Kim and D. J. Park, Unsupervised video object segmentation and tracking based on new edge features, *Pattern Recognition Letters*. **25**(15), 1731–1742 (2004).
- [9] D. Wang, Unsupervised video segmentation based on watersheds and temporal tracking, *IEEE Transactions on Circuits and Systems for Video Technology*. **8**(5), 539–546 (1998).
- [10] B. N. Subudhi, P. K. Nanda, and A. Ghosh, A change information based fast algorithm for video object detection and tracking, *IEEE Transactions on Circuits and Systems for Video Technology*. **21**(7), 993–1004 (2011).
- [11] J. B. Kim and H. J. Kim, Multiresolution-based watersheds for efficient image segmentation, *Pattern Recognition Letters*. **24**(1-3), 473–488 (2003).
- [12] X. He, N. H. C. Yung, K. P. Chow, F. Y. L. Chin, R. H. Y. Chung, K. Y. K. Wong, and K. S. H. Tsang, Watershed segmentation with boundary curvature ratio based merging criterion. In *Proceedings of the 9th IASTED International Conference on Signal and Image Processing*, pp. 7–12 (2007).
- [13] A. Ghosh, B. N. Subudhi, and S. Ghosh, Object detection from videos captured by moving camera by fuzzy edge incorporated Markov Random Field and local histogram matching, *IEEE Transactions on Circuits and Systems for Video Technology*. **22**(8), 1127–1135 (2012).
- [14] B. N. Subudhi, P. K. Nanda, and A. Ghosh, Entropy based region selection for moving object detection, *Pattern Recognition Letters*. **32**(15), 2097–2108 (2011).
- [15] R. O. Hinds and T. N. Pappas, An adaptive clustering algorithm for segmentation of video sequences. In *Proceedings of International Conference on Acoustics, Speech and Signal Processing*, vol. 4, pp. 2427–2430 (1995).
- [16] G. K. Wu and T. R. Reed, Image sequence processing using spatio temporal segmentation, *IEEE Transactions on Circuits and Systems for Video Technology*. **9**(5), 798–807 (1999).
- [17] S. D. Babacan and T. N. Pappas, Spatiotemporal algorithm for joint video segmentation and foreground detection. In *Proceedings of European Signal Processing Conference*, pp. 1–5 (2006).
- [18] E. Y. Kim and S. H. Park, Automatic video segmentation using genetic algorithms, *Pattern Recognition Letters*. **27**(11), 1252–1265 (2006).
- [19] Y. Tsai and A. Averbuch, Automatic segmentation of moving objects in video sequences: A region labeling approach, *IEEE Transactions on Circuits and Systems for Video Technology*. **12**(7), 597–612 (2002).
- [20] S. S. Huang, L. C. Fu, and P. Y. Hsiao, Region-level motion-based back-

ground modeling and subtraction using MRFs, *IEEE Transactions on Image Processing*. **16**(5), 1446–1456 (2007).

- [21] P. M. Jodoin, M. Mignotte, and C. Rosenberger, Segmentation framework based on label field fusion, *IEEE Transactions on Image Processing*. **16**(10), 2535–2550 (2007).
- [22] B. N. Subudhi and P. K. Nanda. Compound Markov random field model based video segmentation. In *Proceedings of SPIT-IEEE Colloquium and International Conference*, vol. 1, pp. 97–102 (2008).
- [23] D. Comaniciu and P. Meer, Mean shift: A robust approach toward feature space analysis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*. **24**(5), 603–619 (2002).
- [24] A. Ghosh, B. N. Subudhi, and L. Bruzzone, Integration of Gibbs Markov random field and Hopfield-type neural networks for unsupervised change detection in remotely sensed multitemporal images, *IEEE Transactions on Image Processing*. **22**(8), 3087–3096 (2013).
- [25] A. Ghosh, N. S. Mishra, and S. Ghosh, Fuzzy clustering algorithms for unsupervised change detection in remote sensing images, *Information Sciences*. **181**(4), 699–715 (2011).
- [26] N. Otsu, A threshold selection method from gray-level histograms, *IEEE Transactions on Systems, Man and Cybernetics*. **9**(1), 62–66 (1979).
- [27] B. N. Subudhi, S. Ghosh, and A. Ghosh, Change detection for moving object segmentation with robust background construction under Wronskian framework, *Machine Vision and Applications*. **24**(4), 795–809 (2013).

Chapter 22

Recent Advances in Remote Sensing Time Series Image Classification

Lorenzo Bruzzone¹, Begüm Demir¹ and Francesca Bovolo²

¹*Dept. of Information Engineering and Computer Science
University of Trento, Trento, Italy*

lorenzo.bruzzone@ing.unitn.it, demir@disi.unitn.it

²*Center for Information and Communication Technology
Fondazione Bruno Kessler, Trento, Italy
bovolo@fbk.eu*

This chapter revises the recent advances in the automatic classification of remote sensing (RS) image time series (images regularly acquired by satellite-borne sensors on the same areas at different times) to update land-cover maps. The availability of up-to-date land-cover maps has significant impacts on several aspects of daily life from the economical, administrative and management point of view. Thus land-cover maps updating by classification of RS images is an hot topic. This is even more true since an increasing number of image time series are being acquired and freely available for a large number of satellite missions (*e.g.*, Landsat archive, ESA Sentinel missions). Land-cover maps can be updated by direct supervised classification of each image in the time series. However, in order to properly train the classifier such an approach requires reliable ground reference data for each available temporal image. In operational scenarios, gathering a sufficient number of labeled training samples for each single image to be classified is not realistic due to the high cost and the related time consuming process of this task. To overcome these problems, domain adaptation (DA) methods have been recently introduced in the RS literature. Accordingly, in this chapter the most recent methodological developments related to DA are presented and discussed by focusing on semi-supervised and active learning approaches. Finally, the most promising methods to define low-cost training sets and to effectively classify RS image time series will be discussed.

22.1. Introduction

The repeat-pass nature of satellite orbits results in multitemporal remote sensing (RS) image series (images taken from the same scene at different time instants). The availability of huge amounts of such data pushed research into the development of data analysis techniques for RS image time series. Time series of RS images can be used for both detecting land-cover transitions occurred on the ground and updating land-cover maps. Both kinds of processes are very important for regularly monitoring the Earth surface. Extracted information is highly relevant to support and improve environmental management policies for both private and public bodies and in several application domains (*e.g.*, administrative, economical, agricultural). Since the number of multitemporal images is increasing day by day and free data access policies are becoming more common (*e.g.*, Landsat Thematic Mapper archive, ESA Sentinel mission), it is important to develop methods to update land-cover maps with both high accuracy and low cost.

Land-cover maps can be updated by automatically classifying each image in the time series, which requires the availability of reference samples for each considered image to train the supervised classifier. However gathering such data is highly costly in terms of human time and effort, and thus also in terms of money. Moreover, often it is impossible from the practical viewpoint to collect reference training information for a given acquisition in the series. As an example, there are remote sensing applications that require the classification of a time series of historical images for which ground truth is not available and can be not collected in a retrospective way. Thus classifying a time series assuming that a sufficient number of ground reference samples is available for each acquisition is unfeasible.

In order to deal with this issue and make it possible to generate a classification map for any desired image for which no prior information is available (target domain), there is a need to find out proper mechanisms. A way is to use reference samples already available for another image acquired on the same scene but at a different time (source domain). The simplest solution would be to use the classifier directly trained on the source domain to classify the target domain. However in most of the cases this does not provide reliable classification results, because source and target domains may differ from each other due to several reasons, such as differences in the atmospheric conditions at the image acquisition dates, different acquisition system state, different levels of soil moisture, changes occurred on

the ground, etc. [1].

To deal with this problem, transfer learning (TL) techniques, and more in detail domain adaptation (DA) methods in TL, have been recently introduced in the RS literature. TL addresses the problem of identifying which knowledge can be transferred and how to transfer it across the domains [2, 3]. Within TL, DA methods aim at classifying the target domain (for which no ground information is available) by exploiting the information available on the source domain, assuming that the two domains may have different, but strongly related, distributions [4]. In the literature, DA is known also as partially supervised/unsupervised learning and is addressed with Semi-Supervised Learning (SSL) methods [5–10] or Active Learning (AL) methods [12–15]. On the one hand, SSL applies a classifier trained on the source domain to the target domain after tuning the parameters according to unlabeled data from the target domain [5–10]. In other words, the information of reference training samples from the source domain is improved by costless unlabeled samples from the target domain to obtain a reliable classifier for the target domain. On the other hand, AL methods aim at improving (from the target domain point of view) the information of the source domain reference samples by iteratively adding the samples selected from the target domain [12–15]. Before inclusion in the training set, these samples should be manually labeled by a human expert. Thus these methods have an associated cost that SSL techniques do not have. In order to reduce the huge cost of collecting large amount of labeled samples, AL methods try to label the smallest possible number of unlabeled samples. This is achieved by selecting for labeling those samples that are the most informative from the target domain viewpoint. Table 22.1 summarizes the main characteristics of the aforementioned learning methods.

In this chapter we initially discuss DA techniques that are proposed in the literature for the classification of RS image time series in the context of SSL and AL. We analyze the critical problems related to different types of learning approaches in the context of DA. Then, we focus our attention on the most recent methodological developments related to: (i) low-cost definition of an effective training set for the target domain; and (ii) target domain classification.

The rest of this chapter is organized as follows. Section 22.2 reviews the DA techniques proposed in the literature for the classification of RS image time series. Section 22.3 investigates the state-of-the-art approaches devoted to low-cost generation of training sets for the target image, while Section 22.4 introduces promising target image classification methods. Sec-

Table 22.1. Main characteristics of the considered learning problems.

Type of Learning	Hypotheses	Objective
Standard Supervised Learning	<ul style="list-style-type: none"> Labeled training data are available for the target domain. 	Exploit available labeled samples to classify the target domain.
Learning Under Domain Adaptation	SSL [5]-[10]	<ul style="list-style-type: none"> Labeled training data are available for the source domain. Costless unlabeled samples from the target domain are in use to tune the parameters of the classifier.
	AL [11]-[14]	<ul style="list-style-type: none"> Labeled training data are available for the source domain. A small number of labeled training samples for the target domain is chosen based on interactions between the classifier and supervisor.

tion 22.5 draws the conclusion of this chapter.

22.2. Domain Adaptation Driven by Semi-Supervised Learning and Active Learning

This section provides a discussion and a review on the use of SSL and AL techniques presented in the literature in the context of DA for classification of remote sensing image time series.

22.2.1. DA in the Context of SSL

DA addressed by SSL aims at exploiting a classifier trained on the source domain for the target domain after tuning its parameters by using unlabeled samples of the target domain [5–10]. For example a DA method (named as a partially unsupervised classification method) is presented in [5]. This method is able to update the parameters of a parametric maximum-likelihood (ML) classifier trained on the source domain on the basis of the statistical distribution of samples in the target domain (for which training

data are not available). This method has been generalized in the context of the Bayes decision rule for cascade classification in [6] in order to exploit the temporal correlation between domains. A further improvement of this approach is proposed in [7] by presenting a multiple cascade-classifier system that is made up of ML and radial basis function neural-network classifiers. SSL-DA-based methods have been developed also in the context of Support Vector Machine (SVM) classifiers. In [9], the initial discriminant function is estimated on the source domain labeled samples. The adaptation to the target domain is achieved by iteratively including in the training set unlabeled patterns from the target domain that have a high probability to be correctly classified. Simultaneously labeled samples of the source domain are gradually removed [9]. A DA method for the binary hierarchical classifier (BHC) is presented in [10]. This method aims at updating the parameters of a BHC classifier trained on the source domain on the basis of the statistical distribution of samples in the target domain. The algorithm can be used when either no labeled training samples are available (unsupervised case) or a small number of training samples exist for the target domain (semisupervised case).

All the above-mentioned methods are defined under two assumptions: (i) the set of land-cover classes that characterizes the target domain should be the same as those included in the source domain, and (ii) the land-cover class statistical distributions should be sufficiently correlated (but not necessarily identical) between the domains. However, in some real RS classification problems these assumptions can not be satisfied due to (i) the possible appearance and/or disappearance of the land-cover classes during time, and (ii) the possible significant differences on the class statistical distributions in the image time series. To overcome the limitations induced by the former assumption, in [8] the sets of classes of the target and the source domains are automatically analyzed in the DA phase by the joint use of unsupervised change detection and Jeffreys-Matusita statistical distance measure. This process results in the detection of classes that appeared or disappeared between the domains. Despite the method presented in [8] can manage possible class differences, the values of the statistical parameters modeling classes propagated from the source to the target domain (*i.e.*, the ones not being changed) are still biased. Thus the latter assumption still needs to be satisfied to achieve satisfactory results.

22.2.2. DA in the Context of AL

DA addressed with AL aims at iteratively selecting the most informative unlabeled samples of the target domain to be included in the training set after manually labeling by a supervisor. The most informative unlabeled samples are the samples that have the lowest probability to be correctly classified by the current classification model, and thus have maximum uncertainty among all unlabeled samples [11]. Differently from SSL, in the context of AL a small number of labeled training samples for the target domain is considered in addition to the labeled samples of the source domain. Accordingly, it is possible to include the information on new appeared classes in the training set via a manual labeling process. In [12] the classifier parameters are initialized by the values estimated on the labeled samples of the source domain. Then the unlabeled samples of the target domain that have the maximum information gain (measured by the KullbackLeibler divergence) are included in the training set of the target domain after manual labeling and the classifier is updated accordingly. In [13] the statistical parameters of a ML classifier are initialized by exploiting the labeled samples of the source domain, and the most informative samples are iteratively selected from the target domain by AL to be added to the training set after manual labeling. During the AL process, the source domain samples that do not fit with the distribution of the classes in the target domain are removed [13]. The abovementioned methods initialize the parameters of the classifier to be applied to the target domain (*i.e.*, the image to be classified) exploiting the labeled samples of the source domain. The adaptation to the target domain starts therefore from biased estimates. If the land-cover class statistical distributions in the source and target domains differ significantly from each other, the adaptation step cannot be fulfilled successfully. In these cases DA methods may provide low classification performance on the target domain.

Recently TL based systems, which are not affected by the differences in classes distribution between the domains, were introduced for automatically updating classification maps by using image time series [14, 15]. The main idea of the these systems is to classify the target domain (\mathbf{X}_2) by defining a strategy that: i) reuses the available information on the source domain (\mathbf{X}_1) (*i.e.*, the source domain training set T_1) by mitigating possible bias effects, and ii) labels the smallest possible number of unlabeled samples from the target domain for optimizing the classification accuracy. To this end, TL systems are defined on the basis of two steps: i) low-cost defini-

tion of a training set (T_2) for the target domain with transfer and active learning methods; and ii) target domain classification. Both the systems in [14] and [15] firstly define an initial training set for the target domain by a transfer learning based on temporal correlation. This is achieved by a change detection based TL (CDTL) strategy without any labeling cost. Then, they optimize the initial training sets by AL strategies that exploit temporal correlation in different ways. The first one consists of a change detection based AL (CDAL) procedure [14]. The CDAL gives priority to the labeling of the samples detected as changed at early iterations (Priority AL), while selecting samples among all unlabeled samples (*i.e.*, changed and unchanged ones) at the remaining iterations (Standard AL). The second one relies on an AL technique defined by conditional entropy that considers the temporal correlation between the target and source domains for the selection of most informative unlabeled samples in the image (see Section 22.3 for further details) [15]. Finally, the target domain is classified by means of the optimized training set. Classification can be performed with or without considering temporal correlation among target and source domains. Figure 22.1 shows the block diagram of the considered architecture that exploits TL and AL for the classification of the target image.

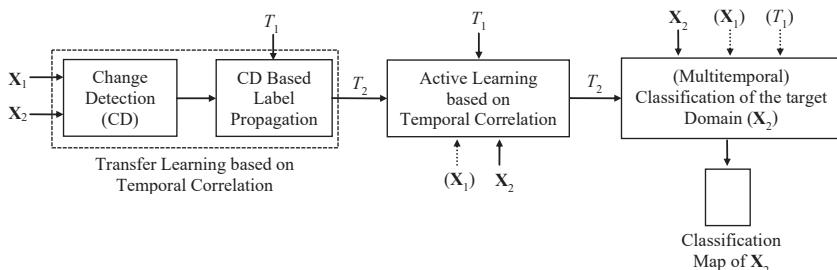


Fig. 22.1. Block diagram of the considered architecture that exploits TL and AL for the classification of the target image (\mathbf{X}_2) based on the source image (\mathbf{X}_1) and training set (T_1). The system exploits the temporal correlation between the considered images (items in brackets represent optional input depending on the specific implementation of the block).

22.3. Generation of a Low-Cost Training Set for the Target Image based on TL and AL

Let $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_P]$ be a time series that consists of P co-registered images acquired at different times on the same scene. Let us assume that at least one of the images in the time-series has a reliable training set. It does not matter whether this is an earlier or older image. The goal is to classify one or more images in the time series even if no reference information is available for them. In order to simplify the mathematical treatment, we consider only a pair of images from the series (*i.e.*, one for the source domain and one for the target domain). Let $\mathbf{X}_1 = \{x_{1,1}, x_{1,2}, \dots, x_{1,B}\}$ and $\mathbf{X}_2 = \{x_{2,1}, x_{2,2}, \dots, x_{2,B}\}$ be two multimodal (multidimensional) images extracted from the time series \mathbf{X} acquired at times t_1 and t_2 , respectively. Both images include C channels and B pixels. Let $(x_{1,j}, x_{2,j})$ be the j -th pair of temporally correlated pixels made up of a pixel $x_{1,j}$ acquired at time t_1 and a spatially corresponding pixel $x_{2,j}$ acquired at time t_2 . Let us assume that the image \mathbf{X}_1 is the source domain for which a reliable training set $T_1 = \{x_{1,j}, y_{1,j}\}_{j=1}^M$ is available. $x_{1,j} \in \mathbf{X}_1$ is the j -th training sample, $y_{1,j} \in \Omega$ is the associated class label, and $M < B$ is the number of training samples. The image \mathbf{X}_2 is the target domain for which a training set T_2 is not available. Let $\Omega = \{\omega_1, \omega_2, \dots, \omega_R\}$ be the set of classes at time t_1 , and $N = \{\nu_1, \nu_2, \dots, \nu_N\}$ be the set of classes at time t_2 . We assume that different sets of classes may characterize the domains (*i.e.*, $\Omega \neq N$). The goal becomes to provide a classification map of the image \mathbf{X}_2 by exploiting both the training samples available on the image \mathbf{X}_1 and the temporal correlation between \mathbf{X}_2 and \mathbf{X}_1 , and by minimizing the labeling costs associated to the definition of T_2 (which at the limit can be null). In this section, we will investigate the most promising methods recently introduced in the RS for the low-cost definition of T_2 . In particular, we will initially present the CDTL approach that defines a reliable training set for the target domain with zero-labeling cost [14, 15]. Then, different AL methods will be introduced that aim at optimizing the training sets obtained by the CDTL [14, 15].

22.3.1. Change-Detection-Driven Transfer Learning

The Change-Detection-driven Transfer Learning (CDTL) aims at defining an initial training set for the image \mathbf{X}_2 (target domain) by taking advantage of the available knowledge from \mathbf{X}_1 (source domain) [13]. The basic idea

behind this choice is that the labels of training samples in T_1 can be considered reliable for \mathbf{X}_2 , and thus transferred, only if the related pixels did not change. Accordingly, at first there is a need of detecting whether changes occurred on the ground between \mathbf{X}_1 and \mathbf{X}_2 , and how many land-cover transitions occurred. As no training set is assumed available for \mathbf{X}_2 , an unsupervised change-detection method should be used. Unsupervised change detection has been widely investigated in the literature [17–19]. Among all the approaches Change Vector Analysis (CVA) technique showed to be a simple yet effective method when dealing with optical passive sensors images [17, 18]. In the CVA technique temporally correlated (thus spatially corresponding) pixels $x_{1,j}$ and $x_{2,j}$ are subtracted to each other in order to build a multispectral difference image \mathbf{X}_D . CVA works under the assumption that multitemporal images are radiometrically correct and co-registered. Such assumptions can be satisfied by proper pre-processing. If after co-registration residual misregistration affects multitemporal data, CVA-based change detection methods robust to this kind of problem can be adopted [22]. The information present in the multispectral difference image is analyzed according to the theoretical framework for unsupervised change detection based on the CVA in polar domain proposed in [18]. According to [18], for simplicity two spectral bands out of C are selected such that the most informative features with respect to the specific considered problem are isolated excluding noisy and misleading spectral channels from the analysis. It is worth noting that, even if the assumption of working with a couple of spectral channels is reasonable in many change-detection problems [17–19], the CVA can be also applied to all spectral channels. In the defined 2-dimensional feature space, the polar representation of multispectral difference image is built on the basis of the magnitude and the direction of spectral change vectors. In this feature space, unchanged pixels are concentrated close to the origin of the polar domain and fall within the *Circle of no-changed* (C_n) pixels, whereas changed pixels fall far from the origin within the *Annulus of changed* (A_c) pixels [17]. The threshold value T that separates C_n from A_c along the magnitude variable ρ can be computed according to any thresholding technique available in the literature [19, 20]. Changed pixels belonging to different land-cover transitions (*i.e.*, different kinds of change) show along the direction variable ϑ in A_c different preferred directions and fall therefore in different *Annular sectors of change* ($S_h, h = 1, \dots, H$, where H is the number of detected kinds of change). Each sector is bounded by a pair of angular thresholds ϑ_{h_1} and ϑ_{h_2} that can be automatically detected according to the method described

in [20] or interactively identified according to a visual analysis of the polar domain. Figure 22.2 shows the decision regions according to the CVA framework.

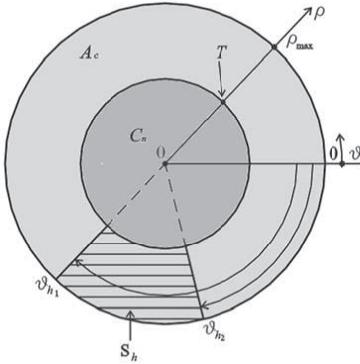


Fig. 22.2. Representation of the regions of interest for the CVA technique in the Polar coordinate system.

Once change detection has been performed, the knowledge related to \mathbf{X}_1 that has a high probability to be reliable also for \mathbf{X}_2 is transferred. Such information is represented by the labels of pixels that are detected as being unchanged, *i.e.*, the ones that fall in C_n . Let $T_1^{UC} = \{x_{1,j}, y_{1,j}\}_{j=1}^R$ be the set of unchanged training pixels at \mathbf{X}_1 , where $x_{1,j} \in T_1^{UC}$ is the j -th training sample and $R(R < M)$ is the number of unchanged training samples. The initial training set T_2 of \mathbf{X}_2 is defined as $T_2 = \{x_{2,j}, y_{1,j}\}_{j=1}^R = \{x_{2,j}, y_{2,j}\}_{j=1}^R$ where $x_{2,j} \in \mathbf{X}_2$ is the j -th initial training sample and $y_{1,j} \equiv y_{2,j}$ is its label transferred from T_1^{UC} . The classes that are represented in T_2 define the set of classes N^{TL} for \mathbf{X}_2 after TL. N^{TL} is by definition a subset of land-cover classes at t_1 (*i.e.*, $N^{TL} \subseteq \Omega$). The labels of all training samples detected as being changed, *i.e.*, the ones that fall in $S_h (h = 1, \dots, H)$, are not transferred.

Unlike other TL techniques presented in the RS literature, the CDTL approach does not require to adapt the classifier parameters of the source domain to the target domain as only sample labels are transferred. Thus it results robust to the class statistical distribution differences between the source and target domains. It is worth noting that although here the CVA technique is considered, the CDTL can be implemented with any unsupervised change detection technique.

22.3.2. Change-Detection-Based Active Learning

The CDTL completely removes from the representation of the problem at t_2 the information about changed pixels, since it does not transfer labels of training samples that fall in $S_h(h = 1, \dots, H)$ assuming that they are unreliable as possibly changed. However, changed pixels are highly important for the map-updating process since they may carry information about possible new classes appeared in \mathbf{X}_2 and/or about different statistical properties of spatially shifted classes (*i.e.*, classes that are already in N^{TL} but appear in a different spatial positions in \mathbf{X}_2 with respect to \mathbf{X}_1). Neglecting this information would lead to unreliable classification of \mathbf{X}_2 . Thus we can state that from the AL viewpoint unlabeled changed samples in \mathbf{X}_2 are potentially highly uncertain (and thus informative) with respect to the classification of \mathbf{X}_2 . Accordingly, the change-detection based AL (CDAL) strategy aims at expanding the training set T_2 obtained by the CDTL by focusing the attention on changed pixels only [14]. In this perspective, pixels among changed samples associated to new classes appeared in \mathbf{X}_2 are the most informative, since they are not represented in the training set T_2 , whereas pixels associated to spatially shifted classes are highly informative only if their spectral signature is different from the one of the pixels of the same class already included in T_1^{UC} . On the opposite, changed pixels that are associated to spatially shifted classes and that have a spectral signature similar to each other are less informative from the AL point of view. The CVA technique chosen for the CDTL is able to identify the presence of different land-cover transitions, whereas it does not provide any information about the new labels assumed by changed pixels in \mathbf{X}_2 (*i.e.*, it is unsupervised). In order to distinguish among the above-mentioned three change cases, the method based on statistical distance measures proposed in [8] is adopted. Let $E = \{\varepsilon_1, \varepsilon_2, \dots, \varepsilon_H\}$ be the set of unknown class labels that changed pixels assume in \mathbf{X}_2 . Note that the number of transitions can be estimated on the basis of the number of annular sectors of changes detected in the polar domain. In order to understand whether $\varepsilon_h \in E$ is already present in the initial training set T_2 (*i.e.*, $\varepsilon_h \in N^{TL} \cap E$) or not (*i.e.*, $\varepsilon_h \notin N^{TL}$) the similarity between the statistical distribution of each $\varepsilon_h \in E$ and that of each land-cover class $\omega_u \in N^{TL} \subseteq \Omega$ present in the initial training set T_2 is computed. Class similarity is measured according to a pairwise Jeffreys-Matusita (*JM*) distance due to its asymptotic behavior. Unlike other distance measures that are unbounded, the *JM* distance reaches saturation to the square root of 2. This behavior makes it easy

to establish a threshold value that defines a high distance (in terms of the Chernoff upper bound to the error probability [25]) of classes [8]. However any other distance measure [23] can be used. Thereby, the *JM* distance JM_{hu} between $\varepsilon_h \in E$ and $\omega_u \in N^{TL}$ can be calculated as:

$$JM_{hu} = \sqrt{2(1 - e^{-B_{hu}})} \quad (22.1)$$

where B_{hu} is the Bhattacharyya distance between the two considered classes and can be calculated as

$$B_{hu} = -\ln \left\{ \int_{X_2} \sqrt{p(X_2|\varepsilon_h)p(X_2|\omega_u)} dX_2 \right\}. \quad (22.2)$$

where $p(X_2|\varepsilon_h)$ and $p(X_2|\omega_u)$ are the class conditional density functions of the random variable X_2 associated to the image \mathbf{X}_2 . Under the assumption of Gaussian distributed classes, (22.2) can be rewritten as:

$$B_{hu} = \frac{1}{8}(\mu_h - \mu_u)^T \left(\frac{\Sigma_h + \Sigma_u}{2} \right)^{-1} (\mu_h - \mu_u) + \frac{1}{2} \ln \left(\frac{1}{2} \frac{|\Sigma_h + \Sigma_u|}{\sqrt{|\Sigma_h||\Sigma_u|}} \right), \quad (22.3)$$

where μ_h and μ_u are the mean vectors of the classes ε_h and ω_u , respectively, Σ_h and Σ_u are their covariance matrices, and T represents the transpose operator.

If for a given $\varepsilon_h \in E$ all computed pairwise *JM* distances are higher than a user defined threshold value Th , we detect the presence of a new class (*i.e.*, $\varepsilon_h \notin N^{TL}$) or of a spatially shifted class (*i.e.*, $\varepsilon_h \in N^{TL}$) with a significantly different spectral signature compared to the same class already present in T_1^{UC} . These are the two situations in which changed pixels are highly uncertain and thus particularly informative for the AL phase. If $\varepsilon_h \in E$ exhibits a small *JM* distance with one of the $\omega_u \in N^{TL} \subseteq \Omega$ classes, a spatially shifted class with spectral signatures similar to those of the same class already present in T_1^{UC} is detected. This procedure is applied until all classes in E have been analyzed. Once this information has been retrieved, the proposed approach applies AL with a mechanism of priority. If classes $\varepsilon_h \in E$ that show a high *JM* distance to all land-cover classes in N^{TL} have been detected, at the first iterations of the AL process the pool of unlabeled samples for \mathbf{X}_2 is made up only of pixels associated to these classes (*i.e.*, changed samples). These pixels are candidate to be either a new class or a spatially shifted class which is not properly modeled by the samples in the training set T_2 . This step is called Priority AL Step. If there is only one unknown class with a high *JM* distance (*i.e.*, $H = 1$), the most uncertain b samples are selected from this class only and labeled

manually to be included in the current training set. In the case there are more unknown classes with high JM distances (*i.e.*, $H > 1$), the most uncertain b/H samples are selected from each class, which results in the selection of b most uncertain samples in total. The manual labeling of the additional b samples solves the ambiguity between spatially shifted classes (*i.e.*, $\varepsilon_h \in N^{TL}$) that have different spectral signatures and new classes ($\varepsilon_h \notin N^{TL}$). Accordingly, in case of $\varepsilon_h \notin N^{TL}$ the set N of land-cover classes at time t_2 is defined as $N = N^{TL} \cup \{\varepsilon_h\}$, whereas if $\varepsilon_h \in N^{TL}$, the set N of land-cover classes is equal to the initial set of classes N^{TL} , *i.e.*, $N = N^{TL}$. In the second part, priority is removed and standard AL is applied including all unlabeled samples in the pool, *i.e.*, both unchanged and changed samples. It is worth noting that if there are no changed

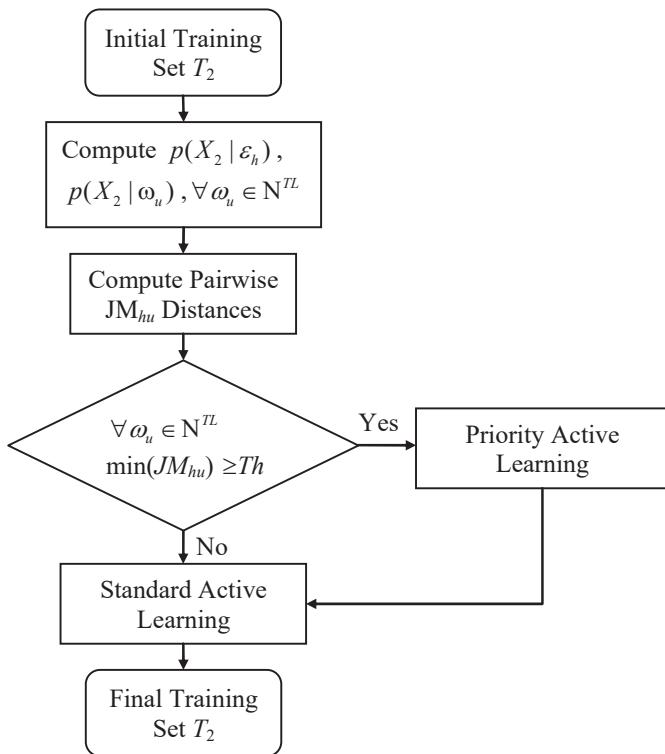


Fig. 22.3. Block diagram of the CDAL.

pixels, the algorithm avoids the Priority AL Step. This is the case in which no new classes are detected and $N \equiv N^{TL}$. Moreover it may happen that the labels propagated from \mathbf{X}_1 are sufficient for a reliable classification of \mathbf{X}_2 . In this case no additional labeled data will be collected for \mathbf{X}_2 . Figure 22.3 shows the block diagram of the CDAL, whereas Fig. 22.4 demonstrates the relationship between N and N^{TL} in the different cases. It is worth nothing that CDAL is independent from the AL technique, and thus can be used with any AL technique presented in the literature.

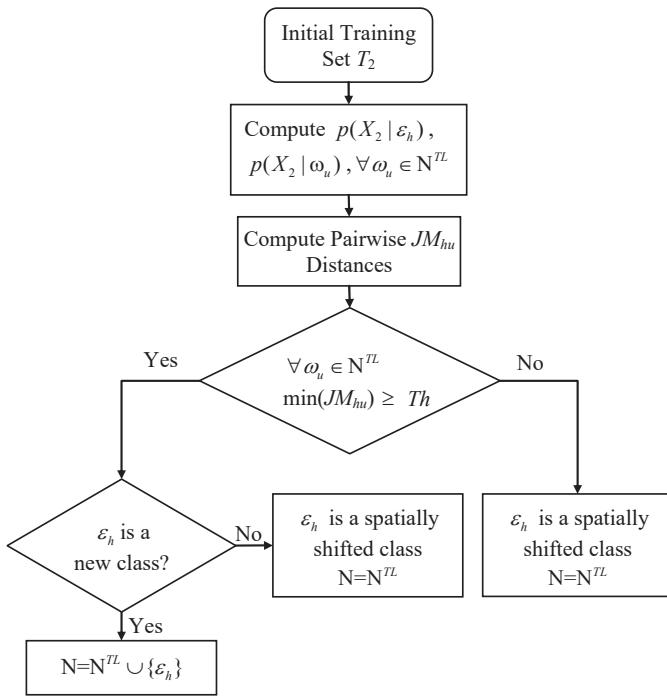


Fig. 22.4. The relationship between the set of classes obtained at the convergence of the CDAL and the set of classes obtained by the CDTL.

22.3.3. Conditional-Entropy-Based Active Learning

The conditional-entropy-based active learning (CEAL) method aims at defining a reliable training set T_2 for \mathbf{X}_2 with a low-cost procedure that takes into account the temporal correlation between \mathbf{X}_1 and \mathbf{X}_2 [15]. In or-

der to apply CEAL we assume that an initial training set is available for the image \mathbf{X}_2 , which can be obtained by adopting the CDTL approach without the need of a labeling process (see Section 22.3.1). The CEAL method models the uncertainty of unlabeled samples by taking into account the temporal correlation between \mathbf{X}_2 and \mathbf{X}_1 , and thus results in a conditional uncertainty criterion. To consider temporal dependence in modeling the uncertainty of samples, the CEAL is defined on the basis of conditional entropy. Let $H(x_{2,j}|x_{1,j})$ be the conditional entropy for the pixel $x_{2,j}$, given the observation $x_{1,j}$, *i.e.*,

$$H(x_{2,j}|x_{1,j}) = - \sum_{\nu_k \in N} P(\nu_k|x_{1,j}, x_{2,j}) \log P(\nu_k|x_{1,j}, x_{2,j}) \quad (22.4)$$

where $P(\nu_k|x_{1,j}, x_{2,j})$ is the probability that the j -th pixel $x_{2,j}$ at t_2 belongs to class ν_k , given the two observations $x_{1,j}$ and $x_{2,j}$. The estimation of $P(\nu_k|x_{1,j}, x_{2,j})$ is a complex task, and therefore the conventional assumption of class-conditional independence in the time domain to simplify the estimations is adopted as in [6, 7]. Under this assumption we can write:

$$\begin{aligned} H(x_{2,j}|x_{1,j}) &= - \sum_{\nu_k \in N} \left[\frac{\sum_{\omega_i \in \Omega} p(x_{1,j}|\omega_i)p(x_{2,j}|\nu_k)P(\omega_i, \nu_k)}{\sum_{\omega_s \in \Omega, \nu_r \in N} p(x_{1,j}|\omega_s)p(x_{2,j}|\nu_r)P(\omega_s, \nu_r)} \right. \\ &\quad \times \left. \log \left(\frac{p(x_{1,j}|\omega_i)p(x_{2,j}|\nu_k)P(\omega_i, \nu_k)}{\sum_{\omega_s \in \Omega, \nu_r \in N} p(x_{1,j}|\omega_s)p(x_{2,j}|\nu_r)P(\omega_s, \nu_r)} \right) \right] \end{aligned} \quad (22.5)$$

where $p(x_{1,j}|\omega_i)$ and $p(x_{2,j}|\nu_k)$ are the single-date class-conditional density functions, and $P(\omega_i, \nu_k)$ is the joint prior probability of having classes ω_i at t_1 and ν_k at t_2 . According to (22.5), $P(\omega_i, \nu_k)$ is the only term that models the temporal correlation between the two domains. A small value of $H(x_{2,j}|x_{1,j})$ shows that the decision of the cascade classifier on the pixel $x_{2,j}$ is reliable, whereas a high value of $H(x_{2,j}|x_{1,j})$ points out that the decision of the cascade classifier is not reliable, and thus the corresponding sample is uncertain (*i.e.*, informative for the classifier). At the first iteration of the CEAL method conditional entropy is estimated according to class statistical parameters evaluated on T_1 and on the initial T_2 obtained by CDTL. Then the training set is enlarged by adding a batch of unlabeled samples with maximum conditional entropy (*i.e.*, those that have the maximum uncertainty among classes) after manual labeling by a supervisor. Once T_2 has been enlarged, the class probability density functions associated to the image \mathbf{X}_2 and the joint prior probability of classes should be

updated accordingly. The joint prior probabilities of classes are derived from the images under investigation. To this end, each image is classified independently from the others by the standard Bayesian maximum posterior probability classifier [23], and then the class transitions are calculated. It is worth noting that the class-conditional density functions at time t_1 are estimated from the available training set T_1 and thus remain fixed during the AL iterations. Once parameters have been updated, the conditional entropy for each unlabeled sample present in the image \mathbf{X}_2 is estimated by (22.5). This process is iterated until convergence, which is achieved when either the values of class parameters do not change from an iteration to the next one, or the upper bound of the cost for labeling samples is achieved (*i.e.*, the maximum possible number of samples is labeled) [15]. Figure 22.5 shows the architecture of the CEAL method.

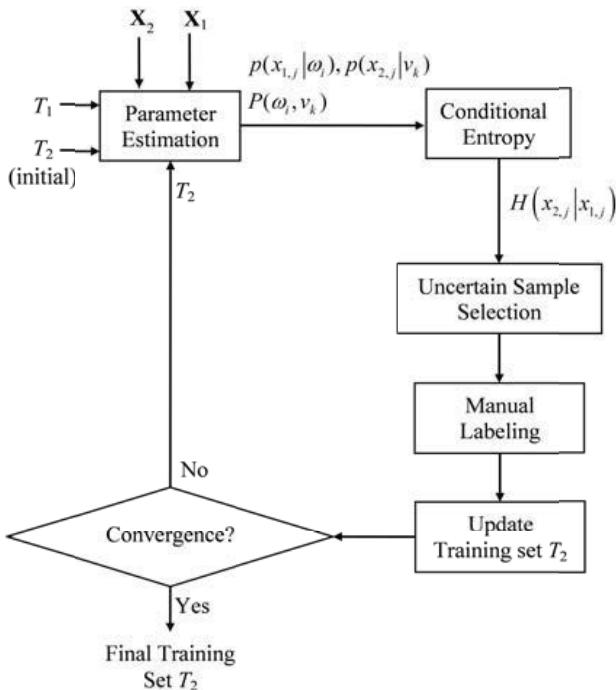


Fig. 22.5. Architecture of the Conditional-Entropy-based Active Learning (CEAL) method.

It is worth noting that the concept of entropy has been already used in the literature for the definition of uncertainty in the context of AL [26, 27]. In [26], marginal entropy is applied to select uncertain training samples for single-date image classification without considering any temporal information. In [27] the concept of multitemporal uncertainty is introduced. Joint entropy is used to select pairs of uncertain multitemporal training samples for the joint classification of multitemporal images in the context of compound classification [27]. The CEAL method significantly differs from [26, 27] due to the fact that it models the uncertainty of unlabeled samples only in the target domain but it also considers the temporal correlation between the target and source domains.

22.4. Classification of the Target Image: Cascade Classification under Bayesian Decision Framework

When the AL process is completed, the target image \mathbf{X}_2 is required to be classified to obtain the land-cover classification map. This is done by training the classifier with the training set defined for \mathbf{X}_2 . To this end, any classification technique presented in the literature can be used. Among several classification techniques, the target domain classification according to the Bayesian cascade decision rule has recently gained a significant interest in RS. The Bayesian decision rule for cascade classification identifies the class label to be assigned to each pixel $x_{2,j} \in \mathbf{X}_2$ by considering the temporal dependence of \mathbf{X}_2 from \mathbf{X}_1 [6, 15], *i.e.*,

$x_{2,j} \in \nu_n$ if

$$\nu_n = \operatorname{argmax}_{\nu_k \in N} \{P(\nu_k | x_{1,j}, x_{2,j})\} = \operatorname{argmax}_{\nu_k \in N} \{p(x_{1,j}, x_{2,j} | \nu_k) P(\nu_k)\} \quad (22.6)$$

where $P(\nu_k)$ is the prior probability of having class ν_k at t_2 . $p(x_{1,j}, x_{2,j} | \nu_k)$ is a mixture density that depends on the distributions of classes at t_1 , and can not be estimated without making this dependence explicit. To overcome this problem, $P(\nu_k | x_{1,j}, x_{2,j})$ can be rewritten by highlighting its dependence on the class labels at time t_1 :

$$P(\nu_k | x_{1,j}, x_{2,j}) = \frac{\sum_{\omega_i \in \Omega} p(x_{1,j}, x_{2,j} | \omega_i, \nu_k) P(\omega_i, \nu_k)}{\sum_{\omega_s \in \Omega} \sum_{\nu_r \in N} p(x_{1,j}, x_{2,j} | \omega_s, \nu_r) P(\omega_s, \nu_r)} \quad (22.7)$$

where $p(x_{1,j}, x_{2,j} | \omega_i, \nu_k)$ is the joint class-conditional density function. The estimation of the statistical quantities in (22.7) is a complex task due to the difficulty in collecting enough training samples for properly modeling

the multitemporal correlation between all possible temporal combinations of classes [6]. Therefore, as often done in practical applications [6, 7], we assume the conventional class-conditional independence in the time domain (*i.e.*, $p(x_{1,j}, x_{2,j}|\omega_i, \nu_k) = p(x_{1,j}|\omega_i)p(x_{2,j}|\nu_k)$) to simplify the estimations process. Under this assumption (22.6) can be rewritten as:

$$x_{2,j} \in \nu_n \text{ if } \nu_n = \left\{ \frac{\sum_{\omega_i \in \Omega} p(x_{1,j}|\omega_i)p(x_{2,j}|\nu_k)P(\omega_i, \nu_k)}{\sum_{\omega_s \in \Omega} \sum_{\nu_r \in N} p(x_{1,j}|\omega_s)p(x_{2,j}|\nu_r)P(\omega_s, \nu_r)} \right\} \quad (22.8)$$

The class parameters at time t_1 can be estimated using the available training samples in T_1 , while the estimation of class parameters at time t_2 and joint prior probabilities $P(\omega_i, \nu_k)$ depend on the availability of the training set T_2 . In the literature, the estimation of these terms has been achieved using two different approaches. The first approach [6] relies on an iterative procedure defined on the basis of the Expectation-Maximization (EM) algorithm [7]. In the iterative EM algorithm, the values of the parameters for the density functions of classes at time t_2 are initialized by considering the corresponding values estimated at time t_1 , whereas joint prior probabilities are initialized by assigning equal joint prior probabilities to each pair of classes (including possible specific constraints on some transitions according to prior information). However, the EM algorithm may not provide accurate estimates when the statistical distributions of classes in the source and target domains significantly differ from each other. As mentioned before, this may happen due to the different acquisition conditions. Such differences may cause the initialization values of the parameters of the density functions for the classes at time t_2 considerably different from the accurate values. The second approach [15] exploits the CDTL and CEAL approaches for the definition of a training set T_2 and thus to the estimation of the mentioned statistical terms are directly achieved by the use of T_2 .

The Bayesian decision rule for cascade classification is very promising as it generates a classification map for an image in the time series using temporal correlation between the images. However, it can be only applied to a pair of images from the time series, and thus ignores the possible information present in the other images of the same time series. To overcome this problem, in [16] a sequential cascade classification method is presented, which extends the standard cascade classification approach to longer time series of remote sensing images. This method assumes that a reliable training set is initially available only for one of the images (*i.e.*, the source domain) in the time series, whereas it is not for the image to be

classified (*i.e.*, the target domain). Then, training sets of all images in the time series included between the source and target domains are obtained by sequentially applying the CDTL method.

22.5. Discussion and Conclusion

This chapter addresses remote sensing time series image classification problem in the critical condition where the available labeled training samples are limited. This operational condition is very common in RS image classification problems, due to the high cost and time related to the collection of labeled samples. To this end, in this chapter we presented methods that aim at classifying an image for which no reference data are available (target domain) by exploiting another image acquired on the same area at a different time for which reference data are available (source domain). In particular, we focused on the domain adaptation methods driven by semi-supervised learning and active learning for updating land-cover classification maps by classifying RS image time series. Then, we reviewed the most recent techniques for generating low-cost training sets for the target domain and discussed the recent trends for the automatic classification of the target image based on the use of temporal correlation among the images acquired on the same area at different times.

To define a low-cost training set for the target domain, we initially illustrated the Change-Detection-driven Transfer Learning (CDTL) method, which propagates the class labels of unchanged samples from the source domain to target domain to define a training set for the target domain. The CDTL offers two main advantages: i) the sample labeling cost in order to populate the training set is zero due to transferring the class labels of the unchanged source training patterns, and ii) the dissimilarity between the class distributions of source and target domains does not affect the proposed method since the original samples of the target domain are directly used to estimate the related classifier parameters (*i.e.*, there is no need to adapt the classifier parameters of the source domain to the target domain, as only the label of samples is transferred).

Afterward, we discussed two state-of-the-art AL approaches that can be applied: (i) to optimize the training set of the target domain defined by the CDTL via labeling a small number of most informative unlabeled samples; and (ii) to mitigate the limitations on the sets of land-cover classes that characterize the two domains. The latter issue is due to the fact that the training set for the target domain (which only contains land-cover classes

shared by the target and source domains due to class-label propagation of unchanged samples in the CDTL) can be enriched with possible new classes during the AL. The first AL method is based on giving a priority to samples detected as being changed during the CDTL procedure. This is because they have a high probability to be most informative among all the unlabeled samples. Therefore this choice allows one to increase rapidly the classification accuracy by labeling few specific unlabeled samples of the target domain, and thus to optimize the classification accuracy. In the remaining iterations priority is removed, and AL is applied in a standard way. The second AL method is based on the conditional entropy, and evaluates the uncertainty (*i.e.*, importance) of samples taking into account the temporal dependence modeled by the joint prior probabilities of classes.

Finally, we discussed the state-of-the-art target domain classification methods. We presented standard and sequential cascade classification methods that take into account the temporal correlation between the images in the time-series.

It is worth emphasizing that updating classification maps in a cost-effective way is becoming more and more important in real applications. This is due to the increased number of time series of remote sensing images. In this context, the methods discussed in this chapter are very promising as they generate a classification map for a generic image in the time series for which no prior information is available, decreasing significantly the cost and effort required for reference data collection. As a final remark, we would like to point out that the investigated methods are general and can be adopted within several application domains in the framework of the analysis of image time-series.

References

- [1] L. Bruzzone, D. Fernandez Prieto and S. B. Serpico, A neural-statistical approach to multitemporal and multisource remote-sensing image classification, *IEEE Transactions on Geoscience and Remote Sensing*, **37**(3), 1350–1359, May (1999).
- [2] S. J. Pan and Q. Yang, A survey on transfer learning, *IEEE Transactions on Knowledge and Data Engineering*, **22**(10), 1345–1359, Oct. (2010).
- [3] E. W. Xiang, B. Cao, D. H. Hu and Q. Yang, Bridging domains using world-wide knowledge for transfer learning, *IEEE Transactions on Knowledge and Data Engineering*, **22**(6), June (2010).
- [4] S. J. Pan, I. W. Tsang, J. T. Kwok and Q. Yang, Domain adaptation via transfer component analysis, *IEEE Transactions on Neural Networks*, **22**(2), 199–210, Feb. (2011).

- [5] L. Bruzzone and D. Fernandez Prieto, Unsupervised retraining of a maximum-likelihood classifier for the analysis of multitemporal remote-sensing images, *IEEE Transactions on Geoscience and Remote Sensing*, **39**(2), 456–460, (2001).
- [6] L. Bruzzone and D. Fernandez Prieto, A partially unsupervised cascade classifier for the analysis of multitemporal remote-sensing images, *Pattern Recognition Letters*, **23**(9), 1063–1071, Jul. (2002).
- [7] L. Bruzzone and R. Cossu, A multiple cascade-classifier system for a robust a partially unsupervised updating of land-cover maps, *IEEE Transactions on Geoscience and Remote Sensing*, **40**(9), 1984–1996, Sep. (2002).
- [8] K. Bahirat, F. Bovolo, L. Bruzzone and S. Chaudhuri, A novel domain adaptation Bayesian classifier for updating land-cover maps with class differences in source and target domains, *IEEE Transactions on Geoscience and Remote Sensing*, **50**(7), 2810–2826, Jul. (2012).
- [9] L. Bruzzone and M. Marconcini, Domain adaptation problems: a DASVM classification technique and a circular validation strategy, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**(5), 770–787, May (2010).
- [10] S. Rajan, J. Ghosh and M. M. Crawford, Exploiting class hierarchies for knowledge transfer in hyperspectral data, *IEEE Transactions on Geoscience and Remote Sensing*, **44**(11), 3408–3417, Jan (2006).
- [11] L. Bruzzone, C. Persello and B. Demir, *Active Learning Methods in Classification of Remote Sensing Images*, in *Signal and Image Processing for Remote Sensing*, 2nd Edition, Ed: Prof. C. H. Chen, CRC Press Taylor & Francis, Chapter 15, 303–323, (2012).
- [12] S. Rajan, J. Ghosh and M. M. Crawford, An active learning approach to hyperspectral data classification, *IEEE Transactions on Geoscience and Remote Sensing*, **46**(4), 1231–1242, Apr. (2008).
- [13] C. Persello and L. Bruzzone, A novel active learning strategy for domain adaptation in the classification of remote sensing images, *IEEE International Geoscience and Remote Sensing Symposium*, Vancouver, Canada, 3720–3723, (2011).
- [14] B. Demir, F. Bovolo and L. Bruzzone, Updating land-cover maps by classification of image time series: A novel change-detection-driven transfer learning approach, *IEEE Transactions on Geoscience and Remote Sensing*, **51**(1), 300–312, Jan. (2013).
- [15] B. Demir, F. Bovolo and L. Bruzzone, Classification of image time series with limited training data, *IEEE Transactions on Image Processing*, **22**(8), 3219–3233, Aug. (2013).
- [16] B. Demir, F. Bovolo and L. Bruzzone, Sequential cascade classification of image time series by exploiting multiple pairwise change detection, *International Conference on Geoscience and Remote Sensing Symposium*, Melbourne, Australia, 3946–3949, (2013).
- [17] A. Singh, Digital change detection techniques using remotely-sensed data, *International Journal of Remote Sensing*, 10, 6, 989–1003, Jun. (1989).
- [18] F. Bovolo and L. Bruzzone, A theoretical framework for unsupervised change detection based on change vector analysis in the polar domain, *IEEE Trans-*

- actions on Geoscience and Remote Sensing, **45**(1), 218–236, Jan. (2007).
- [19] R. J. Radke, S. Andra, O. Al-Kofahi and B. Roysam, Image change detection algorithms: A systematic survey, *IEEE Transactions on Image Processing*, **14**(3), 294–307, Mar. (2005).
 - [20] L. Bruzzone and D. Fernandez Prieto, Automatic analysis of the difference image for unsupervised change detection, *IEEE Transactions on Geoscience and Remote Sensing*, **38**(3), 1171–1182, May (2000).
 - [21] F. Bovolo, S. Marchesi and L. Bruzzone, A framework for automatic and unsupervised detection of multiple changes in multitemporal images, *IEEE Transactions on Geoscience and Remote Sensing*, **50**(6), 2196–2212, (2012).
 - [22] S. Marchesi, F. Bovolo and L. Bruzzone, A context-sensitive technique robust to registration noise for change detection in VHR multispectral images, *IEEE Transaction on Image Processing*, **19**(7), 1877–1889, Jul. (2010).
 - [23] J. A. Richards and X. Jia *Remote Sensing Digital Image Analysis*, 4th ed. Berlin, Germany: Springer-Verlag, (2006).
 - [24] E. Alpaydin *Introduction to Machine Learning*, MIT Press, Cambridge, Massachusetts, London, England, (2004).
 - [25] L. Bruzzone, F. Roli and S.B. Serpico, An extension of the Jeffreys-Matusita distance to multiclass cases for feature selection, *IEEE Transactions on Geoscience and Remote Sensing*, **33**(6), 1318–1321, Nov. (1995).
 - [26] D. Tuia, F. Ratle, F. Pacifici, M. Kanevski and W. J. Emery, Active learning methods for remote sensing image classification, *IEEE Transactions on Geoscience and Remote Sensing*, **47**(7), 2218–2232, Jul. (2009).
 - [27] B. Demir, F. Bovolo and L. Bruzzone, Detection of land-cover transitions in multitemporal remote sensing images with active learning based compound classification, *IEEE Transactions on Geoscience and Remote Sensing*, **50**(5), 1930–1941, May (2012).

Chapter 23

Sensor Selection for E-Nose

Sunil T. T.¹, Subhasis Chaudhuri¹ and M. U. Sharma²

¹*Department of Electrical Engineering
Indian Institute of Technology Bombay, Mumbai, India*

²*Solid State Physics Laboratory, New Delhi, India*

23.1. Introduction

The human olfactory mechanism can discern information about the chemical composition of hundreds of gases and, to an extent, sense the concentration of some of these gases. It is endowed with millions of neural sensors in the nasal cavity which continuously monitor the inhaled air for the presence of any gas molecule. The human brain then processes this information and makes an inference about the presence or absence of gas molecules in the inhaled air [1] [2].

Electronic noses (E-Nose) are designed to mimic human olfactory systems and are modeled after the human nose. A typical E-Nose design consists of a set of gas sensors followed by data acquisition systems and a pattern recognition algorithm. But, unlike the human nose, the E-Noses are generally designed for a specific set of target gases. Gas sensors are analogous to neural sensors and they try to capture the signatures of specific gases for which the E-Nose is designed. When a target gas is exposed to a sensor, some physical parameters of the sensor such as voltage, current or resistance will change. This change in parameter is captured as a suitable electrical signal for further processing by the data acquisition system. The pattern recognition algorithm then makes an inference about the presence or absence of target gases in the captured data.

Several E-Noses have been reported in literature for diverse applications involving detection and classification of target gases in various domains

such as environmental monitoring, classification of food and detection of hazardous gases in industrial environments [3] [4], [5], [6]. However, most of the present day E-Noses are nowhere near human olfactory system in performance as their detection capabilities are severely limited by many aspects such as the number of sensors used, amount of training needed and the computational complexity of the pattern recognition algorithms used [7], [8].

The most critical sub systems in a practical E-Nose are the pattern recognition algorithm and the sensor array. E-Noses have been built with pattern recognition algorithms such as neural networks, clustering and support vector machines [7], [8]. The pattern recognition algorithm uses features derived from gas signatures captured by the data acquisition system. Development of pattern recognition algorithm involves feature selection, training and validation. In general, the sensor data obtained from E-Nose sensors are vector valued and suitable feature extraction methods must be used to handle such data. For any pattern recognition algorithm, training needs a considerable computational time and the performance of the system depends on how well the training data represents different target gases.

The sensitivity and selectivity of the E-Nose depend primarily on the type of sensors used. Several practical E-noses have been designed with sensor arrays built using metal oxide sensors, quartz crystal microbalance sensors, conducting polymers and surface acoustic wave (SAW) devices [9], [10]. The gas sensor array is exposed to target gases for a finite time and during this exposure some physical/chemical properties of the sensors change which are captured as the gas signature. The pattern recognition algorithm will analyze features extracted from gas signatures obtained via data acquisition systems and come to a conclusion about the gas being inhaled.

A sensor to be used in an E-nose sensor array can exhibit different response characteristics to different target gases. The nature of the sensor response depends on the type of sensor, the gas adsorbent material coated on the sensor surface and several other factors. Ideally one would like to have a large number of sensors in the E-Nose to detect more number of target gases as well as to increase the gas detection accuracy. However, due to various physical constraints on E-Noses one is able to embed only a few number of sensors. Some of these constraints include availability of space for mounting a sensor on the E-Nose hardware, isolation among various sensors already mounted and portability of the designed E-Nose. Hence, the number of sensors is often limited.

Given a large number of possible sensors with varying response characteristics to different target gases, an E-Nose designer has to optimally select a finite subset of sensors which provide the maximum classification accuracy when used in conjunction with a pattern recognition algorithm. The sensor selection scheme should look at the signature vectors obtained from different sensors and choose the best combination of a subset of sensors.

This problem can be approached in two ways as in the case of feature selection for machine learning [11].

- Wrapper methods [12]
- Filter methods [13] [14].

In both the methods one collects training data by exposing the target gases to candidate sensors. In wrapper methods, for finding out the suitability of a sensor subset, one trains an appropriate machine learning algorithm using the training data from the sensor subset and evaluate the classification performance by cross validation [15]. In filter methods, one studies the characteristics of training data and based on some characteristics of this data, a sensor subset is chosen directly from the training data without involving any pattern classification. Wrapper methods are dependent on the type of classifier used and may need extensive training and cross validation for arriving at a final sensor set.

On the other hand, filter methods look at the special characteristics of training data and try to choose an optimal subset and can lead to faster selection compared to wrapper methods. This method being independent of the choice of classifier, is capable of absorbing any new pattern classification algorithm without any change in the hardware.

Various filter methods for selection of sensors have been proposed in the literature for different domains such as robotics [16], target tracking [17] and wireless sensor networks [18]. Solutions exist using methods such as convex optimization [19], submodular function maximization [20], [21] and genetic algorithms [22]. However, all these solutions assume that the sensor response is a scalar quantity. The E-Nose sensors are exposed to target gases for a finite time period and the gas signature is obtained over this time period. Each candidate sensor emits a response vector when exposed to a target gas.

23.2. Experimental setup

The block schematic of a typical E-Nose designed for a finite set of target gases is shown in Fig. 23.1. The setup consists of a gas delivery mechanism, a sensor array, some electronics for data acquisition and pre-processing, algorithms for feature extraction and a pattern recognition system. The gas delivery mechanism controls the gas sensing process. It inhales the target gas from the surroundings for a finite period of time and then exhales using a carrier gas. During inhalation the gas delivery system captures gas molecules from the surrounding environment and sends them to the sensor array. The gas molecules are adsorbed on the sensor surface and the nature of these molecules affect the sensor response. During the exhalation the gas delivery system cleans the sensor with some carrier gas. This ensures that all gas molecules adsorbed by the sensors are removed and the E-Nose is ready for the next sensing cycle.

The sensor array in a typical E-Nose will contain 4 to 8 gas sensors. Several sensor technologies have been reported in literature with varying success. Some of them are metal oxide sensors, quartz crystal microbalance sensors, conducting polymers and Surface Acoustic Wave (SAW) devices [10] [9]. Some physical parameter such as the voltage, current or impedance of the sensor will change when the gas delivery mechanism exposes the sensor with a target gas during inhalation. This parameter will return to its normal value when the gas delivery mechanism pumps in carrier gas during exhalation.

The data capture and signal processing section of the E-Nose will collect the change in physical parameter of the sensors into electrical signals and convert them to a suitable data format for further processing by the feature extraction and pattern recognition algorithms.

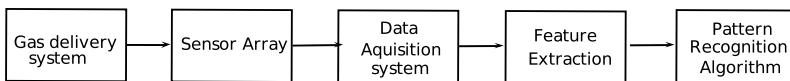


Fig. 23.1. Block diagram of an E-Nose system.

The feature extraction algorithm collects suitable features that represent the gas signature vector. The pattern recognition algorithm is trained with a number of example signatures of target gases prior to actual deployment of the E-nose and is capable of distinguishing different target gases. The E-Nose designer has to carefully select features and then train the pattern

recognition algorithm.

The data used in this article is obtained from an E-Nose which uses a set of SAW sensors. The SAW sensor utilizes the mechanical wave motion which travels across the surface of a certain solid material for gas sensing. The schematic diagram of a typical sensor is shown in Fig. 23.2. There are two inter digital transducers (IDT) on the sensor. The first IDT is used to convert electrical signal to acoustic waves and launch them into the delay line. The second IDT receives the acoustic waves at the other end of the delay line and converts it back to electrical signals. The delay line is coated with some polymers which can adsorb target gases. The nature of adsorption of various target gases can be varied by choosing an appropriate set of polymers. The target gas molecules are exposed to the delay line on the SAW sensor during inhalation. These molecules get adsorbed on the surface thereby changing the velocity of the acoustic wave. Consequently the frequency of the captured electrical signal at the receiving IDT will change. This variation in frequency is dependent on the nature of target gas adsorbed and gas signatures can be identified from this variation.

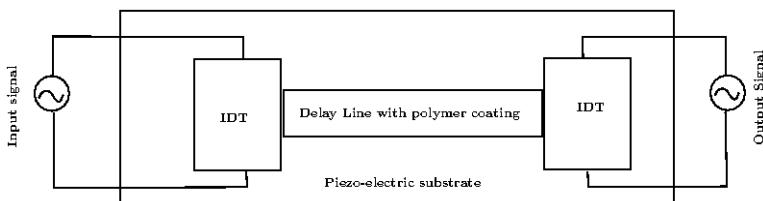


Fig. 23.2. Block diagram of a SAW sensor.

The sensor array used for gas sensing using SAW sensors is shown in Fig. 23.3. The sensors S_1 to S_N are coated with different polymers which can adsorb target gases. The sensor named S_{ref} is used as a reference sensor and is not coated with adsorbent material and hence it does not respond to any of the target gases.

All the SAW resonators in the array are wired as the tuned element of high frequency oscillators. The free running frequency of each oscillator will be determined by the SAW resonator. The frequency of oscillation of S_{ref} is used as a reference. The data acquisition system captures sensor responses from S_1 to S_N as differential variations in frequency with respect to the frequency of S_{ref} . The differential measurement of sensor responses will protect against changes in SAW resonator characteristics with respect to

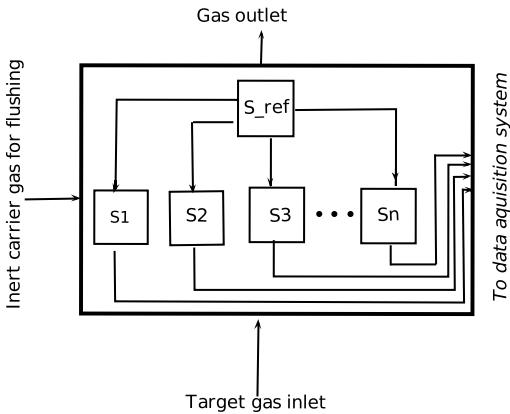


Fig. 23.3. SAW sensor array as mounted on an E-Nose.

variations in ambient temperature, pressure, etc. The experimental setup shown in Fig. 23.3 has been described in detail in [23] and it is claimed to provide good signatures for three target gases, viz dimethyl methylphosphonate (DMMP), dibutyl sulfide (DBS), and dichloropentane (DCP) in air. It is assumed that the placement of sensors and the direction of flow of gas have no effect in the signature response of a gas.

An ideal sensor response to a target gas is shown in Fig. 23.4. There is a base line which corresponds to the carrier gas. The response undergoes a transition from base line at the beginning of exposure until saturation is achieved. The response returns to the baseline when the target gas is flushed out.

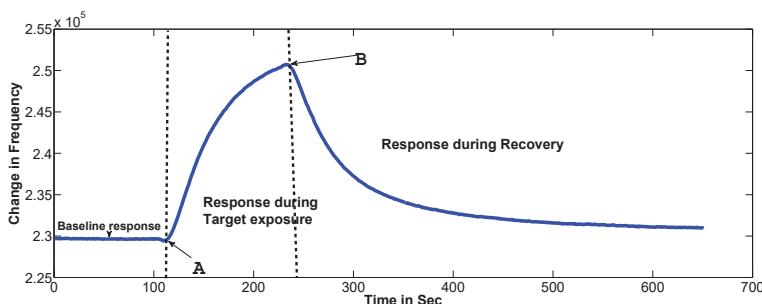


Fig. 23.4. Example of an E-Nose sensor response. A is the time of starting of a sniff cycle and at time B the target gas is flushed out.

Typical responses of two sensors to different gases during exposure are shown in Fig. 23.5. The frequency variation has been captured into an analog signal and is sampled at 2hz. The plot shows the differential variation in frequency with respect the free running frequency of the reference sensor S_{ref} as a function of exposure time. The target exposure was for 1 minute followed by four minutes of purging. Each sensor will be giving a steady state baseline response which corresponds to its free running frequency when exposed to normal air. Upon exposure to a target gas the sensor response changes from its baseline and decreases/increases depending on the nature of the target gas and the polymer coating until it achieves some steady state value. The nature of change in the response will reflect the gas signature of the particular target gas. The response will return to baseline after purging by the carrier gas.

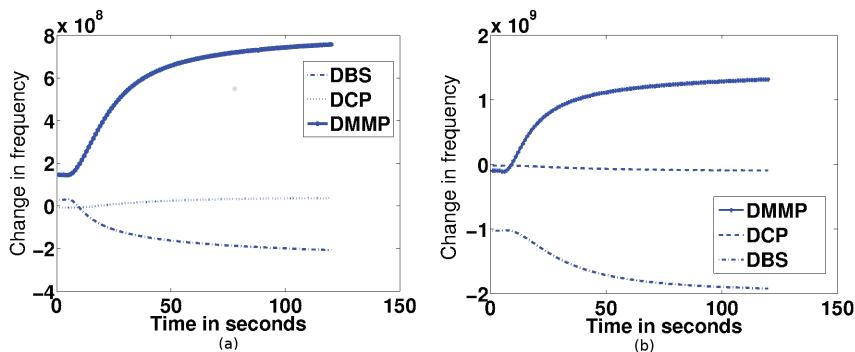


Fig. 23.5. Response of two sensors: (a) sensor 1 and (b) sensor 2 to three different gases.

Our experimental setup has a stock of 12 SAW sensors each coated with derivatives of fluoroalcoholpolysiloxane (SXFA) polymer. The E-Nose has the capability to accommodate 4 sensors apart from the reference sensor. The sensors were numbered from 1 to 12 and then grouped into three. Each group was exposed to three target gases viz DMMP, DCP and DBS under identical conditions. Several response vectors were captured for each target at different concentration levels. At each concentration level, the experiment was repeated several times.

23.3. Gas detection

The choice of a suitable pattern recognition system is one of the most important design decisions an E-Nose designer has to make. This involves selection of appropriate features, selection and training of a classification algorithm from a plethora of available algorithms and finally, validation of the system with suitable test cases. This section explores these issues in the context of data obtained from SAW sensors.

23.3.1. Feature selection

The gas sensors in the E-Nose are exposed to target gases for a finite period of time. The data acquisition system captures the sensor response during the exposure and stores it as a finite dimensional vector. Appropriate features for pattern recognition must be extracted from this vector. The response starts from a baseline and either increases or decreases depending on the target gas until it reaches saturation levels. The nature of the variation is dependent on how the target gas molecules interact with the sensor surface. If we consider the response as a vector in a finite dimensional vector space, the gas signature can be treated as the variation of the vector along certain directions. Hence, principal component analysis (PCA) will be a good choice for feature extraction.

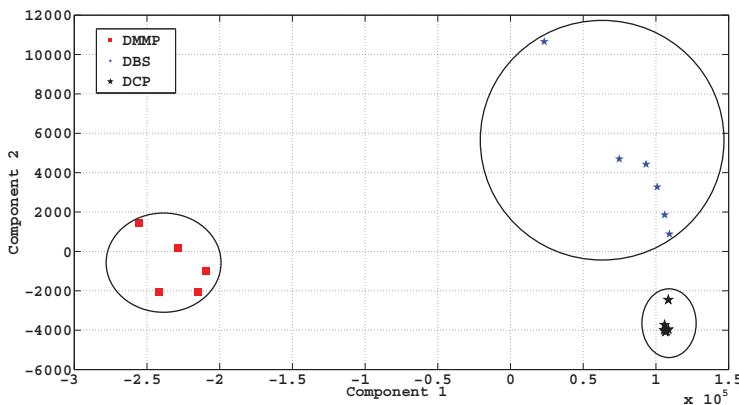


Fig. 23.6. Clustering of target responses when projected onto the first two principal component directions.

Figure 23.6 shows clustering of target responses in the case of a SAW

sensor. The response vectors were projected onto the first two principal components in this plot for easy visualization. It can be seen that these clusters are quite separable, allowing us to detect the gases from their signatures.

Figure 23.7 depicts the distribution of energy along various principal component directions in the case of the above SAW sensor. It may be noted that most of the signal energy is confined to the first few principal directions. Thus the projection of the sensor response to the principal directions with significant eigenvalues can be used as an effective feature for pattern recognition.

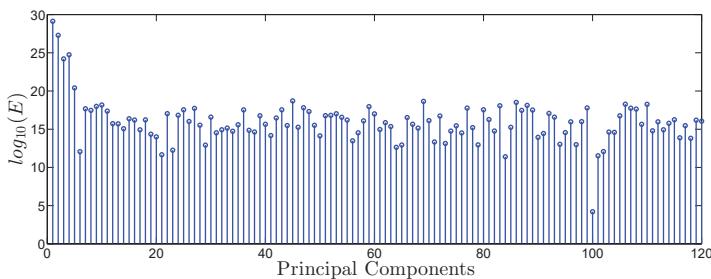


Fig. 23.7. Energy E of the signal along various principal component directions for a SAW sensor.

23.3.2. Choosing the classifier

As already mentioned, a typical E-Nose will have a finite set of sensors. The features extracted from each of these sensors using PCA can be combined to form the final feature vector. For building an effective pattern recognition system we need a good amount of training data representing different target gases. So each target gas must be exposed several times at different concentration levels to obtain the training data.

The choice of the classifier for a given machine learning problem is dictated by several issues such as the number of training samples available, bias variance trade off and the performance criterion. In the case of E-Nose applications the number of training samples are somewhat difficult to collect if the target gases are hazardous and need careful handling. When the training set is small, high bias/low variance classifiers have an advantage over low bias/high variance classifiers, since the latter will over fit [24]. Hence a classifier such as support vector machine can be chosen for E-Nose as the number of training examples is not very large.

23.3.3. Training and validation

The clustering of sensor responses as seen in Fig. 23.6 shows that the data is separable and classifiers can be built using projection of data vectors to the first few principal component directions as features. Since the E-Nose in our experimental setup can accommodate only 4 sensors, the features obtained from all the four sensors in a group are collected together and a combined feature vector is used for training and testing the classifier. The classifiers were validated using leave one out cross validation. The results of cross validation on different sensor sets are summarized in Table 23.1 for initial sensor sets.

Table 23.1. Results of SVM classifier for 3 different mounting of sensors in the E-Nose.

Sensor Set	No of exposures	% of correct classification
S1-S4	131	97
S5-S8	80	96
S9-S12	78	95

The sensor grouping above was done arbitrarily and it may be seen that different groupings give different classification accuracies. So, is there a way to find out which combination gives the best accuracy when we have a large number of possible sensors? The next section explores this problem in detail.

23.4. Sensor selection

Apart from the pattern recognition system, the most important sub-system of an E-Nose is the sensor array. Provided a set of target gases to detect, a single sensor may not have enough sensitivity and selectivity for all the target gases. Hence, a small set of sensors, each having different response characteristics to target gases is employed in the E-nose.

Given a set of target gases and a large number of possible sensors, an E-Nose designer has to optimally choose a small set of sensors such that the data obtained from them will provide the maximum classification accuracy when used along with the pattern recognition system. The choice of sensors must take into consideration the collective response characteristics of all the sensors to various target gases.

One possible solution to sensor selection (M out of N possible sensors) is to run a brute force search which builds a classifier with each of ${}^N C_M$ sensor

combinations. However, each iteration of the above scheme involves training and validation of the classifier with the chosen sensor combination. In a simple test case with 12 possible sensors and the number of target sensors 4, it was seen that a brute force search for sensor combinations with the best classification performance using an SVM classifier took 121 minutes on a PC having Intel core i5 processor and 8GB memory. Out of $^{12}C_4 = 495$ sensor combinations, 89 showed very good classification performance, indicating that several possible combinations may be acceptable. The above computation time will increase with addition of more sensors. The above results were obtained with 80 training examples from 3 different target gases. If the number of training examples are increased, computational requirement of brute force search will automatically increase.

The selection of M sensors from a given set of N possible sensors is a combinatorial optimization problem. In the case of E-Nose applications, two different approaches to the solution can be attempted: filter method and a wrapper method as in the case of feature selection [13], [12], [11] methods in machine learning. The sensor selection problem is different from feature selection since we are trying to choose an optimal set of high level objects which generate a fixed set of features for the classifier.

In the wrapper method, a small subset of data is used to train a chosen classifier and results from the classification is used to select an optimal subset of sensors. Hence, the choice of sensors is dependent on the selection of the specific classifier and the amount of training data and the results may be different if the classifier is changed. In the filter based method, the optimal sensor combination is found by studying the characteristics of sensor responses themselves and the choice of sensors is independent of the classifier to be used.

23.4.1. *Sensor selection using wrapper method*

The results from brute force evaluation of sensors show that there are many possible combinations which give very good results. Any one of them may be chosen for the final nose. Two wrapper based algorithms with significantly reduced computation time than the brute force selection are described below.

23.4.1.1. *Selection based on individual performance*

Assume that we have N available sensors out of which we have to select M sensors that give the best classification performance for a given set of

target gases. The sensors are marked 1 to N for easy identification. We train a classifier using data obtained from all the N sensors and verify it using leave one out cross validation strategy.

The sensors are removed sequentially and N classifiers are trained with data from the remaining sensors. Each of the N classifiers will use data from N-1 sensors. The j^{th} classifier will use data from all sensors except the sensor j . The performance of each classifier is noted in an array P. Now we find out the sensor i whose removal degrades the performance to the maximum by studying the classification performance of the N classifiers we trained. The classification performance degrades because sensor i has important information about the target gas signatures, which is not present in the remaining sensors. Hence, we choose the sensor i to the set of selected sensors. This algorithm can then be repeated on the remaining N-1 sensors. Repetition of the algorithm for M times will give us a set of M sensors that gives the best performance. If at any stage, there are two sensors which show the same maximum degradation in performance when removed, we can select one of them randomly. The algorithm is shown in Algorithm 23.1. Even though this algorithm is a very greedy one, the classifier needs to be run only M iterations to arrive at the final set of sensors. Each iteration consists of one full cycle of training and validation, selecting one sensor from an ensemble.

23.4.1.2. Selection through redundancy elimination

As in the case of previous algorithm, the set of possible sensors are numbered serially from 1 to N. A classifier is trained with all the N possible sensors and the classification performance is evaluated using leave one out cross validation. Next, we train N classifiers with N-1 sensor responses as inputs, by removing sensors 1 to N one by one. The i^{th} classifier will use responses from all the sensors as training data except sensor i ($i=1:N$). The performance of each of these N classifiers is then stored in an array P. The i^{th} element of the array P will contain the performance of a classifier trained without using data from the i^{th} sensor.

If the classification performance does not deteriorate much when the data from the i^{th} sensor is not used, this sensor is redundant and can be removed from the ensemble of target sensors and we are left with N-1 sensors. Next we train N-1 classifiers using response data from a set of N-2 sensor inputs. The classification performance is re-evaluated for each of the N-1 sensors as above and the sensor which exhibits the least degradation in

Algorithm 23.1 Sensor selection based on individual sensor performance.

```

1: Sensor set  $S$  with cardinality  $N$ .  $M$  : no of sensors to be selected
2: Output = List of selected sensors  $A \subset S$ , such that  $|A| = M$ 
3: Initialize an array  $P$  of length  $M$ 
4:  $i \leftarrow 1$ 
5:  $k \leftarrow N$ 
6: loop
7:   if  $i == M$  then
8:     List of optimal sensors are in  $P$ 
9:     return
10:    else
11:      Initialize sensor performance array  $X$  of length =  $k$ 
12:      for  $j \in \{1 \dots k\}$  do
13:        Remove sensor  $j$  from  $S$ 
14:        Compute classification performance using all the remaining sen-
         sors
15:        Store the result in  $X(j)$ 
16:        Replace sensor  $j$  to  $S$ 
17:      end for
18:      Find the sensor whose removal causes maximum reduction in per-
         formance from  $X$ 
19:      Store this sensor in  $P$  and remove it from  $S$ 
20:       $k \leftarrow k - 1$ 
21:       $i \leftarrow i + 1$ 
22:    end if
23:  end loop

```

performance when removed is deleted from the ensemble. This algorithm is repeated until we are left with M sensors. If at any stage, two sensors show the same performance, when removed, one of them is randomly removed and we continue the algorithm with the other. The algorithm is shown as Algorithm 23.2. This method requires $(N-M)$ iterations of classification and performance evaluation. It may be mentioned that this is also a greedy method.

23.4.1.3. Results

As already mentioned, there could be several possible sensor combinations which may offer a good classification accuracy. From the brute force eval-

Algorithm 23.2 Sensor selection through redundancy elimination.

```

1: Sensor set  $S$  with cardinality  $N$ ,  $M$ :no of sensors to be selected
2: Output = List of selected sensors  $A \subset S$ , such that  $|A| = M$ 
3: Initialize an array  $P$  with sensor numbers 1 to  $N$ 
4: loop
5:   if  $sizeof(P) == M$  then
6:     Collect optimal sensors left in  $P$ 
7:     return
8:   else
9:     Initialize sensor performance array  $X$  of length =  $sizeof(P)$ 
10:    for  $j \in \{1 \dots sizeof(P)\}$  do
11:      Remove sensor  $j$  from  $P$ 
12:      Compute classification performance using all the remaining sen-
        sors
13:      Store the result in  $X(j)$ 
14:      Replace sensor  $j$  in  $P$ 
15:    end for
16:    Find the sensor whose removal causes minimum reduction in per-
       formance from  $X$ 
17:    Remove this sensor from  $P$ 
18:  end if
19: end loop

```

ation of sensor combinations several such sets have already been identified. The sensor sets selected by both the wrapper algorithms described above belonged to the set of combinations which provided excellent classification performance. In terms of computation time, the algorithm based on redundancy elimination took 376 seconds to compute the optimal sensor set whereas the algorithm based on individual performance took 96 seconds to compute the same. Both the experiments were done with 12 sensors and 80 training examples obtained from three different gases. The experiment used the same computing setup as that of the brute force method which took 121 minutes. Hence the wrapper methods can be used to reduce the computation significantly.

23.4.2. Sensor selection using filter methods

The wrapper methods described above are dependent on the underlying classifier. Extensive training and validation are done during each iteration

of the wrapper algorithm. Hence, the time taken to evaluate the sensor performance depends heavily on the number of training and test samples. It is sometimes desirable to avoid the classifier and use some property of the training data itself to choose the optimal sensors. Filter methods for sensor selection use this idea to find an optimal sensor set. It also leads to faster computation. Two such filter algorithms are described below. The first algorithm do an exhaustive search using some properties of data whereas the second uses diminishing return property of submodular functions to choose the optimal sensor combination.

23.4.2.1. *Sensor selection as a combinatorial optimization problem*

The accuracy of an E-Nose depends primarily on the success of the underlying pattern recognition algorithm in classifying the set of target gases. If the gas signatures are well separated in feature space, the pattern recognition algorithm can easily identify them. Hence, one desirable feature of a good gas sensor is that it produces well separated gas signatures for all the target gases. However, the sensors may not respond to all the target gases and some of the signatures may not be distinguishable. So we use multiple sensors to overcome this limitation. The combined feature set produced by the selected sensors should be able to classify all instances of target gases. This implies that there must be some dissimilarity in target responses among the selected sensors so that those instances missed out by one sensor must be correctly identified by others. If all the sensors have identical characteristics for a target gas, the classification performance will not improve. So a measure of dissimilarity among the sensors must also be considered while choosing the optimal set. We quantify the above parameters as target separability and sensor similarity. An optimal sensor set is chosen such that the sensors have the maximum target separability and the minimum similarity.

We assume that gas signature obtained from a sensor is a vector in an n dimensional Hilbert space \mathcal{H} where n is the number of sample values obtained by the interfacing circuitry during target exposure. Let $S = \{S_1, S_2, \dots, S_N\}$ be the set of N available sensors and let $G = \{G_1, G_2, \dots, G_K\}$ be the set of target gases to which S is exposed. We must select M sensors optimally from the N available ones. For each sensor in S , assume that P sample responses per gas are available. These P responses are obtained by exposing the target at different concentration levels. Also, at each concentration level, multiple exposures are made. We denote $\bar{X}_{ij}(p)$

as the p^{th} signature vector from sensor S_i when gas G_j is exposed. $\bar{X}_{ij}(p)$ is an $n \times 1$ vector.

We define two parameters, target separability and sensor similarity, from the properties of data with which a quantitative evaluation of the sensor is done. Using these parameters we choose the optimal combination via an exhaustive search.

23.4.2.2. Target separability

Consider the responses of a gas at different concentrations on a particular sensor. Ideally the responses should form a cluster of points around the mean response in the space \mathcal{H} . The location of the cluster in \mathcal{H} will depend on sensor characteristics in relation to the gas. The radius of the cluster will depend on the variations in gas concentration and to a limited extent on the noise introduced in gas delivery and data acquisition system.

For a good classification performance, the responses of chosen sensors to various target gases must form well separated clusters. This implies that the cluster centroids must be far apart in the space H and the cluster radius must be as small as possible. Figure 23.8 shows these ideas pictorially in a two dimensional feature space for three sensors. Points corresponding to two target gases are shown for three different sensors. The target separability of sensor 2 is less compared to sensor 3 since the points corresponding to the two gases are overlapping for sensor 2.

23.4.2.3. Separability index

Separability index of a sensor is defined as a measure of how well separated the data clusters corresponding to various target gases are.

The separability index for a sensor S_i is defined as

$$SI_i = \frac{\text{Between class variance}}{\text{Within class variance}}. \quad (23.1)$$

The within class variance is a measure of how spread out a cluster is from its centroid and it is computed as detailed below. The mean vector response for sensor S_i under exposure to gas G_j is given by

$$\bar{\mu}_i^j = \frac{\sum_{p=1}^P \bar{X}_{ij}(p)}{P}. \quad (23.2)$$

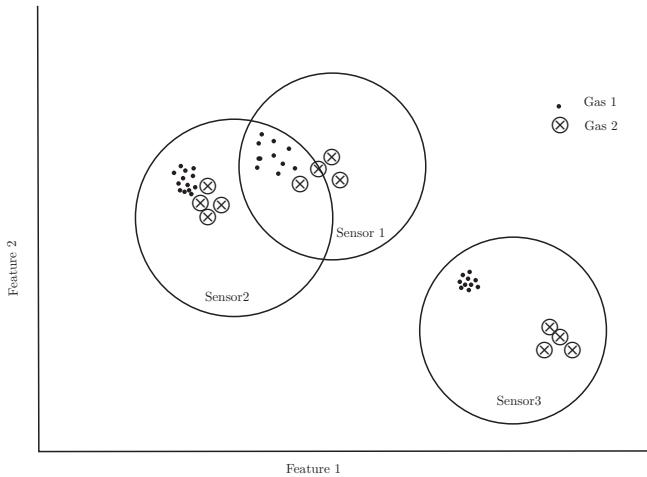


Fig. 23.8. Illustration of separability of target gases.

Let d_{ij}^p be the Euclidean distance of $\bar{X}_{ij}(p)$ from the mean $\bar{\mu}_i^j$.

$$d_{ij}^p = \|\bar{X}_{ij}(p) - \bar{\mu}_i^j\|. \quad (23.3)$$

The mean and variance of d_{ij} for sensor S_i are

$$\mu_{d^{ij}} = \frac{\sum_{p=1}^P d_{ij}^p}{P}.$$

$$\sigma_{d^{ij}}^2 = \frac{\sum_{p=1}^P (d_{ij}^p - \mu_{d^{ij}})^2}{P-1}.$$

The within class variance for sensor S_i is the average of $\sigma_{d^{ij}}^2$ considering all the K target gases.

$$\sigma_i^2 = \frac{\sum_{j=1}^K \sigma_{d^{ij}}^2}{K}. \quad (23.4)$$

For a sensor S_i , the between class variance measures how far away are the cluster centroids corresponding to various target gases located in the space \mathcal{H} . For the sensor i the average response is

$$\bar{\mu}_i = \frac{\sum_{j=1}^K \bar{\mu}_i^j}{K}. \quad (23.5)$$

The Euclidean distance from the mean of individual gas clusters to $\bar{\mu}_i$ can be calculated as

$$d_i^j = \|\bar{\mu}_i - \bar{\mu}_i^j\|. \quad (23.6)$$

A measure of between class variance of sensor S_i is provided by variance of d_i^j .

$$\mu_d = \frac{\sum_{j=1}^K d_i^j}{K}. \quad (23.7)$$

$$\sigma_{di}^2 = \frac{\sum_j^K (d_i^j - \mu_d)^2}{K-1}. \quad (23.8)$$

For each element of S we can calculate separability index as

$$SI(S_i) = \frac{\sigma_{di}^2}{\sigma_i^2} \quad \forall S_i \in S. \quad (23.9)$$

23.4.2.4. Similarity between sensors

If two sensors have very similar responses to all the target gases, no additional information regarding the target will be obtained by including both of them to the final set. So it is desirable to add a measure of dissimilarity among sensors to the final selection criteria. Let $Y = \{\bar{\nu}_1^j, \bar{\nu}_2^j, \dots, \bar{\nu}_N^j\}$ be the set of normalized mean response vectors for target gas T_j on sensor set S where $\bar{\nu}_j^j = \frac{\bar{\mu}_j^j}{\|\bar{\mu}_j^j\|}$. A good measure of similarity of the response of sensor S_p to the response of S_q when exposed to G_j is given by the inner product $\langle \nu_p^j, \nu_q^j \rangle$.

For each sensor in the sensor set S we define a similarity measure with the rest of sensors for each gas G_i .

$$C^j(p, q) = \langle \nu_p^j, \nu_q^j \rangle, \quad p, q \in \{1, \dots, N\}. \quad (23.10)$$

Since there are K target gases, average similarity matrix is

$$C = \frac{\sum_{j=1}^K C^j}{K}. \quad (23.11)$$

23.4.2.5. Selection of optimal sensor combination

Given N candidate sensors, there are $r = \binom{N}{M}$ distinct sensor combinations with cardinality M . Our task is to find out the optimal combination from them which gives the maximum classification accuracy. The elements of the optimal subset must have a good separability index and they must be dissimilar from one another. Let T be the set of all subsets of S with cardinality M .

$$T = \{T_i \mid T_i \subset S, |T_i| = M\} \quad \text{with } |T| = r. \quad (23.12)$$

$$T_i = \{t_{i1}, t_{i2}, \dots, t_{iM}\} \quad \forall t_{ib} \in S, \quad b \in \{1 \dots M\}. \quad (23.13)$$

The solution to the following optimization problem will yield the optimal sensor set.

$$\hat{T}_i = \arg \max_{T_i} (C_1 + \lambda C_2), \quad (23.14)$$

where λ is a weight factor and C_1, C_2 are the components of the cost function corresponding to separability and dissimilarity.

The component of the cost function indicating separability is computed as the sum of separability of each of the sensors.

$$C_1 = \sum_{m=1}^M SI(t_m). \quad (23.15)$$

Each of the t_{ib} in Eq. (23.13) maps to a unique $S_d \in S$ denoted by a $t_{ib} \rightarrow S_d, \quad b \in \{1 \dots M\}$. Define an index set I

$$I = \{i_b \mid b \in \{1 \dots M\}\}. \quad (23.16)$$

The dissimilarity component of the cost function is then computed as

$$C_2 = \sum_m \sum_{n < m} (1 - |C(m, n)|) \quad m, n \in I. \quad (23.17)$$

where C is the similarity matrix defined in Eq. (23.11).

The optimization equation defined in Eq. (23.14) can be solved for obtaining the optimal set of sensors.

23.4.2.6. Results

The 12 SAW sensors used for this experiment are numbered *a* to *l*. Table 23.2 shows the computed separability indices as defined by Eq. (23.9) for the sensors. As seen from the table the sensor numbered *l* has the highest separability. Table 23.3 shows the similarity matrix for the sensors as computed by Eq. (23.11).

Table 23.2. Computed separability indices for the given target gases for various sensors.

Sensor No	Separability index
a	0.5936
b	0.2321
c	0.1200
d	0.1010
e	0.2545
f	0.2699
g	0.1433
h	0.2885
i	0.0264
j	0.1343
k	0.7514
l	2.2721

Table 23.3. Computed similarity measures among all 12 SAW sensors. Those with similarity above 0.9 have been highlighted in the table.

Sensor	a	b	c	d	e	f	g	h	i	j	k	l
a	1	-0.9986	-0.9946	-0.9467	0.1843	0.2799	0.9637	-0.7100	-0.8631	-0.9749	-0.9846	-0.6795
b	-0.9986	1	0.9893	0.9333	-0.2143	-0.2704	-0.9512	0.6804	0.8686	0.9731	0.9787	0.6937
c	-0.9946	0.9893	1	0.9726	-0.1187	-0.3165	-0.9829	0.7670	0.8266	0.9652	0.9852	0.6220
d	-0.9467	0.9333	0.9726	1	0.0496	-0.3969	-0.9781	0.8792	0.6950	0.8925	0.9396	0.4538
e	0.1843	-0.2143	-0.1187	0.0496	1	-0.5601	0.0422	0.3980	-0.3219	-0.2267	-0.1556	-0.3804
f	0.2799	-0.2704	-0.3165	-0.3969	-0.5601	1	0.3426	-0.5465	-0.1095	-0.2217	-0.2679	0.0931
g	0.9637	-0.9512	-0.9829	-0.9781	0.0422	0.3426	1	-0.8120	-0.7882	-0.9481	-0.9790	-0.5556
h	-0.7100	0.6804	0.7670	0.8793	0.3980	-0.5465	-0.8120	1	0.3823	0.6259	0.7081	0.1203
i	-0.8631	0.8686	0.8266	0.6950	-0.3219	-0.1095	-0.7882	0.3823	1	0.9371	0.8868	0.9262
j	-0.9749	0.9731	0.9652	0.8925	-0.2267	-0.2217	-0.9481	0.6259	0.9371	1	0.9910	0.7731
k	-0.9846	0.9787	0.9852	0.9396	-0.1556	-0.2679	-0.9790	0.7081	0.8868	0.9910	1	0.6909
l	-0.6795	0.6937	0.6220	0.4538	-0.3804	0.0931	-0.5556	0.1203	0.9262	0.7731	0.6909	1

There are $\binom{12}{4} = 495$ distinct sensor combinations and a brute force approach to solution of Eq. (23.14) is feasible. The value of the cost function as computed using Eq. (23.14) is plotted in Fig. 23.9. The sensor combinations are numbered from 1 to 495 according to the value of cost arranged in the descending order for the purpose of illustration. For the above plot the chosen value of $\lambda = 100$. It is seen that there are several candidate combinations which give a good accuracy in recognizing a target gas and

all those combinations had their costs quite high in the plot in Fig. 23.9. Any of these combinations can be chosen for the final E-nose. The combination which gives the maximum cost is selected to the final sensor set for the E-Nose.

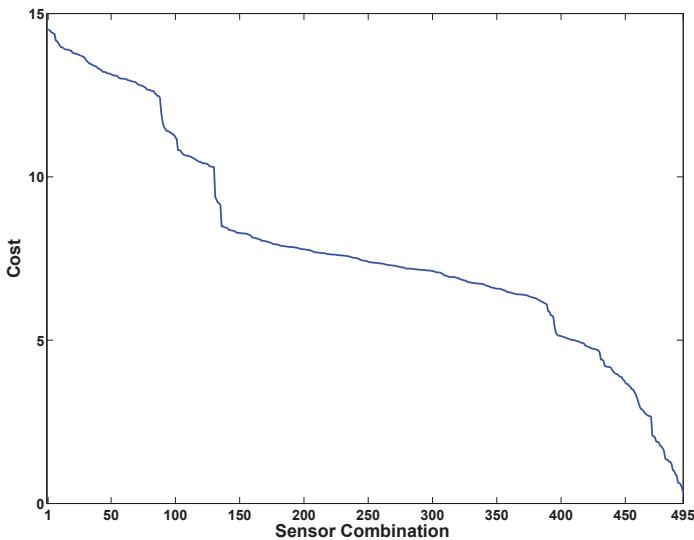


Fig. 23.9. Plot of the computed cost function for all possible quadruplet combinations of candidate sensors arranged in the descending order of magnitude.

The most important advantage of this method over wrapper methods is that this algorithm does not require a predefined classifier to select the optimal set of sensors and can be much faster than the wrapper method when the number of training examples are more. The computation time for the proposed method was 52 seconds on a PC having Intel core i5 processor and 8GB memory. However, the approach is brute force in nature and can still consume a large time depending on the number of sensors and the amount of training data available.

23.4.2.7. Sensor selection using submodular function maximization

Next we formulate a novel algorithm for sensor selection using maximization of submodular functions over a uniform matroid. Let the set of target gases and available sensors be $\{G\}$ and $\{S\}$, respectively. Assume that

the cardinality of G is K and that of S is N . Consider a utility function $U(G_i, A)$ that quantifies the usefulness of a sensor subset $A \subset S$ in correctly identifying $G_i \in G$ while doing classification. If the probability of finding the gas G_i is $P(G_i)$, we define a sensing quality function [25] as

$$F(A) = \sum_{i=1}^M P(G_i)U(A|G_i). \quad (23.18)$$

Several algorithms for sensor selection can be designed by choosing different utility functions [21]. The probability density function $P(G_i)$ can be estimated from the data or can be given as a prior.

However, there are several practical constraints, such as the cardinality or the covering on the subset A . These constraints can be modeled as an additional cost function $C : 2^S \rightarrow R$. In the case of E-Nose sensor selection, this additional constraint is the number of sensors M in the array. Hence sensor selection problem is reformulated as

$$\begin{aligned} A^* &= \arg \max_A F(A) \\ &\text{subject to } |A| = M. \end{aligned} \quad (23.19)$$

The solution to Eq. (23.19) will depend on the characteristics of the sensing quality function. To solve Eq. (23.19), the sensing quality function $F(A)$ is formulated as a submodular function and we maximize it over a uniform matroid. Even though maximization of submodular functions over uniform matroids is NP hard, there are good approximation algorithms which work in polynomial time [26]. The following sections describe the mathematical preliminaries of submodular functions and matroids.

23.4.2.8. Submodular set functions

Submodular functions [26], [27] are set functions which yield diminishing returns when we add more elements to the argument of such functions. Consider a ground set V with a power set 2^V and a function H which maps all $A \in 2^V$ to real numbers

$$H : 2^V \rightarrow R. \quad (23.20)$$

H is said to be a submodular function if for all $A, B \subseteq V$

$$H(A) + H(B) \geq H(A \cup B) + H(A \cap B). \quad (23.21)$$

Equivalently, a set function is submodular if for all $A \subseteq B \subseteq V$ and $j \in V \setminus B$

$$H(A \cup j) - H(A) \geq H(B \cup j) - H(B). \quad (23.22)$$

Equation (23.22) implies that the incremental gain in H obtained by adding a new element to a smaller set is more than what we get if the same element is added to a bigger set which contains the smaller set. If the expressions given in Eq. (23.21) is an equality the function is said to be a modular function. The following properties of submodular functions are used in later sections [28].

Property 1. Sum of two submodular functions is submodular.

Property 2. Sum of a modular function and a submodular function is submodular.

23.4.2.9. Matroids

A matroid [29], [30] is an abstraction generalizing the notion of linear independence to combinatorial objects. A matroid is defined as an ordered pair $\mathcal{M} = (V, I)$ where V is a finite set and I is a collection of subsets of V which satisfy the following properties.

- a. The empty set $\Phi \in I$.
- b. I is hereditary.
 $\forall B \in I, A \subseteq B \rightarrow A \in I$.
- c. I satisfies independent set exchange property.
 $\forall A, B \in I, |A| < |B| \rightarrow \exists x \in B \setminus A, A \cup x \in I$.

If $\mathcal{M} = (V, I)$ is a matroid, the members of I are called an independent set and V the ground set of \mathcal{M} . A maximal independent set is called a basis of the matroid. A uniform matroid on V is defined by taking every k -element subset of I to be a basis. The optimization problem defined in Eq. (23.22) can be treated as maximization of the sensing quality function $F(A)$ over a uniform matroid. Rank of the underlying matroid will be a good choice for defining the utility function defined in Eq. (23.21).

The rank of $A \subseteq V$ is given by the rank function $r(A)$ of the matroid, which has the following properties.

- a. The value of the rank function is always a non-negative integer.
 - b. For any $A \subset V, r(A) \leq |A|$.
 - c. For any two subsets $A, B \subset V, r(A \cup B) + r(A \cap B) \leq r(A) + r(B)$.
- The rank of a matroid is a submodular function.
- d. For any set A and an element $x, r(A) \leq r(A \cup \{x\}) \leq r(A) + 1$.
- Hence, the rank is a monotonic function.

23.4.2.10. Maximization of submodular functions

The algorithm for maximization of a submodular function is given in Algorithm 23.3 [31]. This algorithm uses the diminishing returns property shown in Eq. (23.22) for computing the optimal sensor to be added to the target set at each iteration as discussed in the next section.

Algorithm 23.3 Sensor selection using maximization of submodular functions over uniform matroids.

```

1: Input =  $F$  a submodular function on sensor set,  $M$ : No of sensors to be
   selected
2: Output = List of selected sensors  $A \subset S$ , such that  $|A| = M$ 
3: begin
4:    $A \leftarrow \{\phi\}$ 
5:   for  $j = 1$  to  $M$  do
6:     for  $s \in V \setminus A$  do
7:        $\delta_s \leftarrow F(A \cup s) - F(A)$ 
8:     end for
9:      $S^* \leftarrow \underset{s \in V \setminus A}{\operatorname{argmax}} \delta_s$ 
10:     $A \leftarrow A \cup S^*$ 
11:   end for
12: end

```

This algorithm, even though greedy, can provide strong theoretical guarantees. Nemhauser *et al.* [26] have analyzed the problem of maximization of a submodular function over a uniform matroid and showed that this greedy algorithm is a tight $e/(1-e)$ approximation algorithm for the problem.

The lazy evaluation scheme proposed in [32] can further improve the performance of the algorithm. The lazy evaluation utilizes the property that the incremental gain $\delta_s \leftarrow F(A \cup s) - F(A)$ is non decreasing in nature. δ_s in the current iteration can be compared with the incremental gains of all sensors from previous iteration and if the current δ_s is the largest, that specific sensor can be added to the final set.

23.4.2.11. Sensor selection using submodular function maximization

Let us take a gas $G_i \in \{G\}$. The training data for target gas G_i is collected by exposing the sensors in S to gas G_i several times at different concen-

tration levels. During each exposure, the data acquisition system in the E-Nose samples sensor responses and collects a k element vector for each sensor. We treat each sensor response as a point in a k dimensional space. A set of k orthogonal basis vectors can be formed from principal component analysis for each sensor. The mean response of each sensor to a gas can then be approximated as the linear combination of the first v significant principal components.

Let $\{\nu_1^j, \nu_2^j, \dots, \nu_N^j\}$ be the mean signature vectors for sensors $\{S_1 \dots S_n\}$ when exposed to gas G_j . When we select a sensor to the final set to be used in the E-Nose array, the following properties are desirable:

- The chosen sensor should not be identical to any other sensor already in the array in terms of response to a gas.
- The sensor must be sensitive to all the target gases.
- The sensor must produce a distinct response for each of the target gases.

The utility function used in Eq. (23.18) must capture all the above required characteristics and for a set of sensors A when exposed to gas G_i it is defined as a combination of three functions which try to assimilate the above ideas.

$$U(A|G_i) = R(A|G_i) + \lambda L(A|G_i) + \gamma Q(A). \quad (23.23)$$

The functions $R(A|G_i)$, $L(A|G_i)$ and $Q(A)$, defined below, try to capture the above mentioned desired properties for a set of target sensors A . λ and γ are relative weights.

The function $R(A|G_i)$ tries to capture the similarity between sensor responses when exposed to G_i . For defining $R(A|G_i)$ consider a pair of sensors S_l and S_k with signature vectors ν_l^i and ν_k^i . If the vectors ν_l^i and ν_k^i are linearly dependent, it is assumed that they are similar in characteristics and adding S_l and S_k together to the final sensor set will not improve the classification accuracy. We can extend this argument to more than two sensors. We define $R(A|G_i)$ as the rank of the matrix $[\nu_r^i \dots \nu_m^i]$, where $\nu_r^i \dots \nu_m^i$ are the mean signature vectors for $S_r \dots S_m \in S$ when exposed to G_i . $R(A|G_i)$ is an indicator of the dissimilarity among elements of sensor subset $A = \{S_i \dots S_m\} \subseteq S$ when exposed to gas G_i . It was shown in Section 23.4.2.9 that rank is a submodular function.

$L(A|G_i)$ is defined as the sum of L_2 norms of signature vectors in A .

This function tries to quantify the sensitivity of the sensor to gas G_i

$$L(A|G_i) = \sum_{n=1}^{|A|} \|A_n\|. \quad (23.24)$$

It is easy to show that $L(A|G_i)$ is a modular function. Finally, the function $Q(A)$ tries to quantify how separable the sensor responses are for different target gases obtained from the set A . Let T_{S_i} be a matrix obtained by concatenating the responses for all the target gases from $S_i \in A$, ν_i^1, \dots, ν_i^M . Let $r(T_{S_i})$ be the rank of the matrix T_{S_i} .

$$Q(A) = \sum_{S_i \in A} r(T_{S_i}). \quad (23.25)$$

$Q(A)$ being a rank function is a submodular in nature. The sensing quality function described in Eq. 23.18 can be formulated with a uniform prior probability $P(G_i)$ as

$$F(A) = \sum_{i=1}^M (R(A|G_i) + \lambda L(A|G_i) + \gamma Q(A)). \quad (23.26)$$

The above function is submodular by property 1 and 2 of submodular functions described in Section 23.21. The optimization problem defined in Eq. (23.19) is the maximization of a submodular function over a uniform matroid. Hence, algorithm 23.3 can be used for efficient selection of sensor.

23.4.2.12. Experimental results

The algorithm described above was validated using data collected from the experimental setup detailed in Section 23.2. Principal component analysis was done for each gas for all the sensors. The mean response of a particular target gas on a sensor was then approximated as the linear combination of its projections on the first two principal component directions. This approximated mean response was used to evaluate the sensing quality function formulated in Eq. (23.26).

The sensing quality function in Eq. (23.26) is the sum of three functions: two rank functions $R(A|G_i)$, $Q(A)$, and a norm $L(A|G_i)$. The norm for each gas $L(A|G_i)$ can be directly calculated from its mean response. For calculating the rank functions $R(A|G_i)$ and $Q(A)$ the following procedure was used.

For a set of sensors A , a matrix T was formed by concatenating the respective sensor responses. The rank of T is calculated as below. First, the

condition number of the matrix T is calculated. If the condition number is above some threshold θ , the rank r of the matrix T is taken as the value of r for which $|\lambda_1/\lambda_r| \leq \theta$. λ_i 's are the singular values of the matrix T arranged in descending order. If the condition number is below the threshold θ , the rank of T is directly computed.

The growth of the sensing quality function for several different choices of subsets of sensors are shown in Fig. 23.10. The sensors are numbered a to l as already mentioned in Section 23.4.2.1. Due to the submodularity property, the growth of this function is non-decreasing in nature in all cases. A large growth in this function is an indication that the sensor being added has some different sensing quality compared to the already chosen sensors and can potentially contribute to improve the classification accuracy. It may also be noted that the value of the sensing quality function for the optimal sensor set always predominates values of other sets with the same cardinality. The plots also suggest that the first two choices of subsets $\{f, g, i, k\}$ and $\{a, g, d, e\}$ offer very similar sensing quality values and both choices should yield nearly similar classification accuracies. The plot also shows that sensors f and a have very similar sensor responses and sensor g

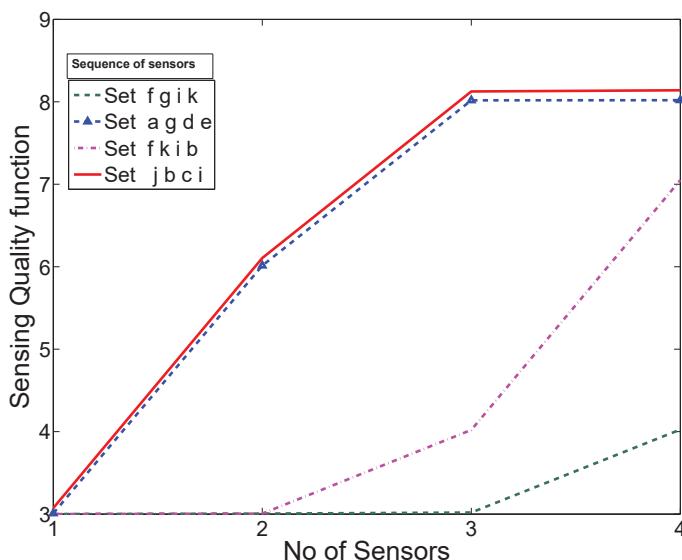


Fig. 23.10. Growth of sensing quality function for four different choices of subsets of sensors.

is quite different in response from either sensor f or a .

It was observed that several sensor combinations gave nearly 100% accuracy and for all those combinations the sensing quality function was showing much higher values compared to those combinations with a lower accuracy. As expected both the subsets $\{f, g, i, k\}$ and $\{a, g, d, e\}$ were found to offer the same classification accuracy. The algorithm will choose the set showing the maximum sensing quality function. The SVM classifier built with features extracted from this set showed the maximum accuracy.

23.5. Conclusions

Four different algorithms have been proposed for optimal selection of E-Nose sensors. In general, for a limited set of target gases, there will be many excellent sensor combinations. All these algorithms are able to choose one of them. The wrapper algorithms are greedy in nature and are dependent on the choice of the classifier. If the amount of training data is very large, wrapper methods can take a very long time to find out the optimal solution as training and cross validation are involved in each iteration of the algorithm. The solutions provided by the algorithms may also be dependent on the characteristics of classifier.

The first filter algorithm proposed in this work, based on data separability and sensor similarity, uses an exhaustive search strategy. If the number of available sensors is very large, the exhaustive search can also take a large time. It is typically suitable for situations when the number of available sensors are less. The algorithm based on submodular function maximization is faster than all other algorithms and is optimal when we have a large number of training data from a number of sensors. Table 23.4 summarizes the computation time needed for the four proposed algorithms with E-Nose data obtained from 12 saw sensors. The experiments were done with a PC having a Pentium Core i5 CPU and 8Gb RAM using Matlab® programming environment. The wrapper methods used support vector machines as the underlying classifier. Table 23.4, along with the fact that the sensors chosen in all the four cases provide very similar accuracy, substantiate our claim that the submodular function maximization method is the most appropriate technique for sensor selection.

Table 23.4. Comparison of computation times of the proposed algorithms.

Algorithm	Time
Redundancy elimination	376 sec
Individual performance	96 sec
Optimization using similarity and separability	52 sec
Submodular function maximization	26 sec

Acknowledgment

The authors would like to thank Solid State Physics Laboratory, New Delhi, India for providing the data used in this article.

References

- [1] A. Keller and L. B. Vosshall, Better smelling through genetics: mammalian odor perception, *Current opinion in neurobiology*. **18**(4), 364–369 (2008).
- [2] J.-B. Watelet and P. V. Cauwenberge, Applied anatomy and physiology of the nose and paranasal sinuses, *Allergy*. **54**, 14–25 (1999).
- [3] O. Canhoto and N. Magan, Electronic nose technology for the detection of microbial and chemical contamination of potable water, *Sensors and Actuators B: Chemical*. **106**(1), 3 – 6 (2005). ISSN 0925-4005. doi: 10.1016/j.snb.2004.05.029.
- [4] R. Dutta, K. Kashwan, M. Bhuyan, E. Hines, and J. Gardner, Electronic nose based tea quality standardization, *Neural Networks*. **16**(5), 847–853 (2003).
- [5] R. E. Baby, M. Cabezas, and E. W. de Reca, Electronic nose: a useful tool for monitoring environmental contamination, *Sensors and Actuators B*. **149** (1), 1–17 (2005).
- [6] H. Zhang and J. Wang, Detection of age and insect damage incurred by wheat, with an electronic nose, *Journal of Stored Products Research*. **43**(4), 489–495 (2007).
- [7] H. T. Nagle, R. Gutierrez-Osuna, and S. S. Schiffman, The how and why of electronic noses, *Spectrum, IEEE*. **35**(9), 22–31 (1998).
- [8] T. C. Pearce, S. S. Schiffman, H. T. Nagle, and J. W. Gardner, *Handbook of machine olfaction: electronic nose technology*. John Wiley & Sons (2006).
- [9] K. Arshak, E. Moore, G. Lyons, J. Harris, and S. Clifford, A review of gas sensors employed in electronic nose applications, *Sensors Review*. **24**(2), 181–198 (2004).
- [10] D. James, S. Scott, Z. Ali, and W. O'hare, Chemical sensors for electronic nose systems, *Microchimica Acta*. **149**(1), 1–17 (2005).
- [11] Y. Saeys, I. Inza, and P. Larrañaga, A review of feature selection techniques in bioinformatics, *Bioinformatics*. **23**(19), 2507–2517 (2007).
- [12] R. Kohavi and D. Sommerfield. Feature subset selection using the wrapper

- method: Overfitting and dynamic search space topology. In *Proceedings of the first international conference on knowledge discovery and data mining 1995*, pp. 192–197 (1995).
- [13] I. Guyon and A. Elisseeff, An introduction to variable and feature selection, *The Journal of Machine Learning Research*. **3**, 1157–1182 (2003).
 - [14] D. Koller and M. Sahami, Toward optimal feature selection, *Stanford InfoLab* (1996).
 - [15] T. T. Sunil and S. Chaudhuri, Detection of target gases and optimal selection of saw sensors for e-nose applications. In *Proceedings of Conference on Advances In Robotics*, AIR '13, pp. 13:1–13:6, ACM, New York, NY, USA (2013). doi: 10.1145/2506095.2506132. URL <http://doi.acm.org/10.1145/2506095.2506132>.
 - [16] G. E. Hovland and B. J. McCarragher, Dynamic sensor selection for robotic systems. In *Proceedings of IEEE International Conference on Robotics and Automation, 1997.*, vol. 1, pp. 272–277 (1997).
 - [17] V. Isler and R. Bajcsy, The sensor selection problem for bounded uncertainty sensing models. In *Proceedings of the 4th international symposium on Information processing in sensor networks 2005*, p. 20 (2005).
 - [18] H. Rowaihy, S. Eswaran, M. Johnson, D. Verma, A. Bar-Noy, T. Brown, and T. La Porta, A survey of sensor selection schemes in wireless sensor networks. In *Defense and Security Symposium 2007*, pp. 65621A–65621A (2007).
 - [19] S. Joshi and S. Boyd, Sensor selection via convex optimization, *IEEE Transactions on Signal Processing*. **57**(2), 451–462 (2009).
 - [20] M. Shamaiah, S. Banerjee, and H. Vikalo, Greedy sensor selection: Leveraging submodularity. In *49th IEEE Conference on Decision and Control 2010*, pp. 2572–2577 (2010).
 - [21] A. Krause, *Optimizing sensing: theory and applications*. PhD thesis, Carnegie Mellon University (2008).
 - [22] L. Yao, W. A. Sethares, and D. C. Kammer, Sensor placement for on-orbit modal identification via a genetic algorithm, *AIAA Journal*. **31**(10), 1922–1928 (1993).
 - [23] A. Nimal, U. Mittal, M. Singh, M. Khaneja, G. Kannan, J. Kapoor, V. Dubey, P. Gutch, G. Lal, K. Vyas, and D. Gupta, Development of hand-held saw vapor sensors for explosives and cw agents, *Sensors and Actuators B: Chemical*. **135**(2), 399 – 410 (2009). ISSN 0925-4005.
 - [24] G. Forman and I. Cohen, Learning from little: Comparison of classifiers given little training. In *Knowledge Discovery in Databases: PKDD 2004*, pp. 161–172. Springer (2004).
 - [25] R. A. Howard, Information value theory, *IEEE Transactions on Systems Science and Cybernetics*. **2**(1), 22–26 (1966).
 - [26] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, An analysis of approximations for maximizing submodular set functions, *Mathematical Programming*. **14**(1), 265–294 (1978).
 - [27] S. Fujishige, *Submodular functions and optimization*. vol. Annals of Discrete mathematics 58, Elsevier (2005).

- [28] L. Lovász. Submodular functions and convexity. In *The State of the Art Mathematical Programming*, pp. 235–257. Springer (1983).
- [29] H. Narayanan, *Submodular functions and electrical networks*. vol. 54, *Annals of Discrete Mathematics*, Elsevier (1997).
- [30] J. G. Oxley, *Matroid theory*. Oxford university press (2006).
- [31] A. Krause and D. Golovin. Submodular function maximization. In eds. L. Bordeaux, Y. Hamadi, P. Kohli, and R. Mateescu, *Tractability: Practical Approaches to Hard Problems*. Cambridge University Press (2012).
- [32] M. Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, pp. 234–243. Springer (1978).

Chapter 24

Understanding the Usage of Idioms in the Twitter Social Network

by Koustav Rudra, Abhijnan Chakraborty, Niloy Ganguly, Saptarshi Ghosh

Department of CSE

Indian Institute of Technology Kharagpur, India

Max Planck Institute for Software Systems, Germany

Department of CST

*Indian Institute of Engineering Science and Technology
Shibpur, India*

To help users find popular topics of discussion, Twitter periodically publishes ‘trending topics’ (trends) which are the most discussed keywords (e.g., hashtags) at a certain point of time. Inspection of the trends over several months reveals that while most of the trends are related to events in the off-line world, such as popular television shows, sports events, or emerging technologies, a significant fraction are *not* related to any topic / event in the off-line world. Such trends are usually known as *idioms*, examples being #4WordsBeforeBreakup , #10thingsIHateAboutYou , etc. We perform the first systematic measurement study on Twitter idioms. We find that tweets related to a particular idiom normally do not cluster around any particular topic or event. There are a set of users in Twitter who predominantly discuss idioms – common, not-so-popular, but active users who mostly use Twitter as a conversational platform – as opposed to other users who primarily discuss topical contents. The implication of these findings is that within a single online social network, activities of users may have very different semantics; thus, tasks like community detection and recommendation may not be accomplished perfectly using a single universal algorithm. Specifically, we run two (link-based and content-based) algorithms for community detection on the Twitter social network, and show that the idiom oriented users get clustered better in one while topical users in the other.

24.1. Introduction

Twitter is now considered more of an ‘information network’ than a social network [1, 2] and almost the entire focus of the research community has been on ‘topical’ content in Twitter, such as tweets / hashtags related to sports or technology or emergency situations in the off-line world [3, 4]. However, a closer inspection of the Twitter *trending topics* (‘trends’ in short) – keywords periodically declared by Twitter as being the most discussed at that point in time – indicates some exceptions to this view, and provides the motivation for the present study.

We collected US trends over a duration of 10 months (January – October, 2014) using the Twitter API at 15-minute intervals. This gave about 18,500 distinct trending topics during this period. We then developed a classifier *Odin*,* using which we classified the trends into multiple categories such as sports, entertainment, technology etc. – these broad categories were identified by human volunteers (details in Section 24.3). Table 24.1 shows the distribution of the trends collected during the 10-month period, in the different broad categories. While most of the categories are topical and related to events in the off-line world, it can be observed that a special category, known as *idioms* regularly becomes trending. An idiom, as defined by Romero *et al.* [5], is a keyword representing a conversational theme on Twitter, consisting of a concatenation of at least two common words, which does *not* include names of people, places or music albums, and so on. Examples of idioms include #4WordsBeforeBreakup, #11ThingsAboutYou, and evidently these are not related to any topic or event in the off-line world.

Table 24.1. Percentage of Twitter trends collected over ten months, and classified into nine different categories that were identified by human volunteers (details in Section 24.3). Also given are few examples of trends.

Category	%	Example trends
Entertainment	33%	#5sonsonKiis, #IWishICould, #Austinonidol
Sports	30%	#argentinavsholanda, #lakers, #bravsger
Idioms	9%	#WhenIWasATeenager, #FactsaboutMe, #Ifwedate
Technology	8%	#iphone6, #galaxy4, AppleWatch, ios8
Politics	5%	#tcot, #pjnet, #obama, #gaza
Business	5%	#amazon, #AlibabaIPO, #FedReserve
Religion	3%	#EidMubarak, #jesus, #citrt
Health	2%	#Ebola, #Who, #breastcancer
Others	5%	#garlicparmpizza, #filipino, cheesecake, pizza is healthy

* Named after the God of Wisdom according to Norse mythology; details in Section 24.3.

It is intriguing that though idioms are not related to any event / topic in the off-line world, yet millions of users discuss idioms on online forums like Twitter. This raises the question whether their dynamics as well as the users discussing such trends are similar to those of the topical counterparts. To understand the dynamics, we collected tweets related to hundreds of idioms and the users who discuss them, and conducted a detailed measurement study. We find that the tweets containing idioms are mainly conversational in nature; for instance, they hardly contain URLs. On investigating the users who post the tweets (the idiom-users), we find that they are mostly general and active Twitter users, as opposed to being popular experts / celebrities who usually drive topics such as politics and entertainment. The idiom-users maintain close friendships among themselves and interact on diverse issues with their friends. Thus, the study unfurls that hidden within the *information network* of Twitter, there is a *social network* of users who regularly have “non-topical” conversations among themselves.

Such an inference has far-reaching implications. It essentially means that multiple dominant dynamics are present in the same social network – so the standard tasks like community detection, recommendation, and so on, *cannot* be done using a one-parameter-fits-all approach. An algorithm to identify (recommend) topical groups might fail to identify (recommend) idiom-users. To test this proposition, we run two community detection algorithms – one identifying topical groups, proposed by Bhattacharya *et al.* [3] and the other, Infomap [6] which detects communities using link structure. We find that the idiom-users are well identified by Infomap while the topical groups are better identified by the algorithm proposed by Bhattacharya *et al.* [3]. This establishes that different approaches for tasks such as clustering may have different utilities in a heterogeneous online social network.

The rest of this chapter is organized as follows. Section 24.2 describes prior work related to hashtags in the Twitter social media, and user-groups in social networks. Section 24.3 explains the procedure for hashtag classification, while Section 24.4 describes the dataset used for the study. Section 24.5 compares the idioms and the topical trends (e.g., trends related to sports, entertainment, politics) with respect to how they are discussed over the Twitter social media, while Section 24.5.2 characterizes the users who are predominantly interested in memes and in topical trends. Finally, Section 24.6 concludes the chapter.

24.2. Related Work

The present study focuses on classifying hashtags and identifying users who actively participate in idioms, and understanding the social behaviour of the groups of these users. We briefly discuss some prior studies on classification of hashtags in Twitter, and some sociological theories on group-formation among users.

Classification of hashtags in Twitter: Many existing studies on trending topics in Twitter have focused on classification of the trends [7, 8], whereby the presence of idioms [8] is identified. However, there has been little effort in analyzing the characteristics of idioms, and of the users who post the idioms. To our knowledge, the only prior study which attempted to compare idioms with trends related to events in the off-line world is by Naaman *et al.* [9]. Different kind of features like content feature, interaction feature, time-based feature etc. are used to classify the trends. They compared ‘exogenous’ and ‘endogenous’ trends, where the exogenous trends are those related to events occurring outside the Twitter universe, while the endogenous trends are the ones which existed within Twitter (i.e., idioms). However, they did not analyze the users who discuss such idioms, and did not attempt to analyze the diffusion patterns of various trends.

The only work which discussed propagation vis-a-vis different categories of tweets was by Romero *et al.* [5] who analyzed the reasons behind the variation in propagation of popular hashtags in Twitter. They chose hashtags related to different categories such as sports, music, television shows, politics in order to identify their propagation pattern. They showed that hashtags on politically controversial topics are adopted more on repeated exposures, while idioms are propagated by simple contagion method. In this work, we try to propose a lightweight hashtag classification scheme based on publicly available knowledge sources like AlchemyAPI, Freebase etc.

User groups in social networks: Several theories have been proposed in sociology and behavioral sciences to explain the formation of groups among users in a social network [10, 11]. One of the most well-known theories on group formation, both in off-line and online societies, is the *common identity and common bond theory* [12, 13], according to which, the attachment of users to a particular group can be either identity-based or bond-based. In identity-based groups, people join the group due to their interest in a

well-defined common theme (topic) shared by the members. On the other hand, attachment in *bond-based* groups is driven by personal social relations (bonds) among the members, and may be characterized by the absence of any central theme / topic of discussion within the group. According to this theory, bond-based groups tend to have higher reciprocity among the members, whereas interactions in identity-based groups are generally not directly reciprocated. Also, the topics of discussion in bond-based groups tend to vary widely and cover multiple subjects, while in identity-based groups, they tend to be related to the common group theme.

There have been several attempts to distinguish between bond-based and identity-based groups, especially in the online world. For instance, Sassenberg *et al.* [14] classified chats among users on a text-based communication platform into two categories – on-topic chats which are on a common topic (identity-based) and off-topic chats where people chatted on a variety of topics (bond-based). Grabowicz *et al.* [15] differentiated between the identity-based and bond-based groups in the Flickr social network. More recently, Bhattacharya *et al.* [3] identified a large number of topical groups in Twitter, comprising of users who are experts or seekers of information on various topics, and showed that these groups are essentially identity-based. In our work, we explore the nature of the groups among the idiom-users, and find that they reveal bond-based characteristics.

24.3. Classification of Trends

In order to perform a large scale study on idioms and the trends related to topics / events in the off-line world, we built an automatic classifier *Odin*, to distinguish particular trends based on whether they are idioms or related to some topic. Some prior studies [7, 8] also attempted to classify trends (not necessarily into the same categories found by *Odin*), utilizing the textual contents of the tweets containing the trends. However, tweets (restricted to 140 characters) often contain informal language and abbreviations which potentially results in lower classification accuracy [8]. Hence, we adopt a different approach that combines both tweets and related web documents and uses several web-based knowledge engines to perform the classification. *Odin* classifies a given trend following the steps presented below.

24.3.1. Preprocessing

Segmentation: Trends often consist of multiple words [5] either in form of hashtags or multi-word phrases. Recognizing these distinct words might be easy for multi-word phrases, but it is very challenging for hashtags. In some hashtags, the distinct words are joined using a CamelCase style, i.e., by capitalizing the first letter of each word (e.g., #WorldCupSoccer, #IsMisuseOfLawNotACrime). But in other cases, all the words are in lowercase or uppercase, and are just juxtaposed without any separation signs (e.g., #everythingididntsay, #selfieforlucky).

Since it is important to identify the individual words which make up a trend in order to understand its topic, trends need to be segmented into the component words. Similar to the approach followed by Berardi *et al.* [16], Odin formulates the hashtag splitting problem as a compound word segmentation problem. There might be multiple combinations of word components which can create a compound word. To choose the most probable sequence of component words given a word compound, Odin follows a modified version of the Viterbi Algorithm [17]. Viterbi algorithm uses a model of word distribution to calculate the most probable character sequence forming a word. Odin computes the word distribution model from Google n -gram corpus (available at <https://books.google.com/ngrams>). Then given a trend, Odin segments the trend into its constituent words based on the calculated probability estimates of multiple candidates and by selecting the most probable one.

Categorization of related web documents: Odin searches different Web search engines (e.g., Google, Bing) with the segmented trend, to get a large set of web-pages relevant to the given trend. Additionally if the tweets containing the trend have URLs, they become another source for getting related web-pages.[†] For a given trend, Odin collects all the web-pages pointed from the tweets and returned by the search engines; and then a set of category keywords are extracted for these collected web-pages using the NLP-based AlchemyAPI web service (www.alchemyapi.com). Table 24.2 shows the different webpage categories identified by AlchemyAPI.

Entity extraction and categorization: Sometimes the trend contains names of people, organizations or locations (e.g., #EMABiggestFansJustin-

[†]URLs leading to social media sites like Facebook, Twitter, Instagram, are ignored, since these pages usually do not have much content to help the topic categorization process.

Table 24.2. Taxonomy of web-pages identified by AlchemyAPI.

Category	Broad domains
Arts and Entertainment	Drawing, Sculpting, Movies, Music, etc.
Business	Business, Finance, SEC filings, etc.
Computers and Internet	Computers, Internet, Mobile Phones, etc.
Culture and Politics	Political news and Culture / Society-related issues.
Gaming	Computer Games, Video Games, Board Games
Health	Medications, Treatments, Diseases, etc.
Law and Crime	Legal and Crime-related News and Discussion.
Religion	Religious News and Discussion.
Recreation	Recreational Activities (Boating, etc.)
Science and Technology	Physics, Astronomy, etc.
Sports	Sports News and Commentary.
Weather	Weather News and Discussion.

Bieber) detecting which might give a clear idea on the category of the trend. Similarly, the web documents and the tweets associated with a particular trend have many such named entities present in them. Odin extracts such entities using AlchemyAPI and then queries Freebase (www.freebase.com) to know the ‘notable type’ of such named entities (e.g., according to Freebase, notable type for ‘Justin Bieber’ is ‘/music/artist’). Table 24.3 shows the mapping between the notable type categories and the trend classes identified by Odin.

Table 24.3. Different notable type categories returned by Freebase; they are grouped into the different categories identified by human volunteers.

Topic	Freebase notable type categories
Entertainment	music, media_common, film, tv, fictional_universe, award, broadcast, opera, celebrities, comedy, digicams, fashion, exhibitions, visual_art, comic_books
Sports	sports, soccer, olympics, american_football, baseball, cricket, basketball, ice_hockey, games, boxing, tennis, skiing, bicycles
Technology	computer, engineering, spaceflight, conferences, meteorology, physics, chemistry, biology, astronomy, geology
Politics	government, military, royalty
Business	finance, industry, business, zoos, amusement_parks, aviation, rail
Religion	religion
Health	medicine
Other	periodicals, location, food, travel, book, education

24.3.2. Classification

At the end of preprocessing steps, for a given trend, Odin computes the distribution of webpage categories and the notable types from the set of related web documents and named entities. Treating these distributions and

fraction of web search results pointing to social media sites (like Twitter, Facebook, Tumblr etc.) as features, Odin uses a Support Vector Machine (SVM) classifier with Radial Basis Function kernel to classify a particular trend into one of the 9 categories shown in Table 24.1.

Training Data Preparation: For creation of training data, three human volunteers (regular users of Twitter, who are not authors of this paper) were asked to manually inspect 700 distinct trends collected during the first two weeks of January 2014 (along with tweets containing these trends), and classify the trends into different categories. The volunteers identified the *nine broad categories* shown in Table 24.1, such as Entertainment, Sports, Technology, Idioms (following the definition of idioms in Romero *et al.* [5]). Out of the 700 trends, all three volunteers agreed upon a particular category for 575 trends. We created the training data considering this unanimous categorization as the ground truth.

Classification Performance: Standard 10-fold cross validation on the data of the 575 trends showed that Odin attains 77.15% accuracy in predicting trend categories, which is good considering that it is a complex nine-class classification task.

24.4. Dataset

Since most of the Twitter trends were related to the three topics *entertainment*, *sports*, and *technology* (see Table 24.1), we decided to focus on idioms and trends related to these three topics; the trends related to any of these three topics are collectively referred to as '*topical trends*'. For each of the trends belonging to the four selected categories, we collected as many tweets containing the trend as possible using the Twitter search API. To get a better understanding about the trends, in our analysis as presented in later sections, we used only those trends for which we were able to collect more than 30,000 tweets. To maintain uniformity across categories, we finally selected a set of 150 trends related to each of the categories (the actual distribution is stated in Table 24.1).

For each of the 600 selected trends, we further collected detailed statistics about all the users (including their profile details, social links and recently posted tweets) who posted a tweet containing any of the selected trends. Table 24.4 summarizes the statistics of the data collected for the trends of the four categories.

Table 24.4. Statistics of data collected for the study.

Property	Idiom	Sports	Entertainment	Technology
Number of trends	150	150	150	150
Total #tweets containing the trends (millions)	6.205	6.787	6.967	6.105
Mean #tweets per trend	41,369	45,257	46,455	40,721
Total #distinct users posting the trend (millions)	2.74	2.71	1.90	1.75
Mean #distinct users per trend	18,315	18,098	12,725	11,705

24.5. Comparing Idioms and Topical Trends

In this section, we compare how idioms and topical trends (which are related to topics / events in the off-line world) are discussed in the Twitter social network, and the users who discuss them frequently.

24.5.1. How trends are discussed in Twitter

We first analyze how the trends of different categories are, in general, discussed in Twitter. For a given trend t , we consider all tweets containing t , and measure what percentage of these tweets contain other hashtags (apart from t itself), and URLs.

Figure 24.1 shows mean values of the percentage of tweets containing other hashtags and URLs, where the mean values are computed over all trends of a particular category. Statistical measures like two sample KS-test and Mann-Whitney U test with significance level 0.05 show that there is a significant difference in the distribution of the mean values among the four categories. Expectedly, we find that the topical trends are much

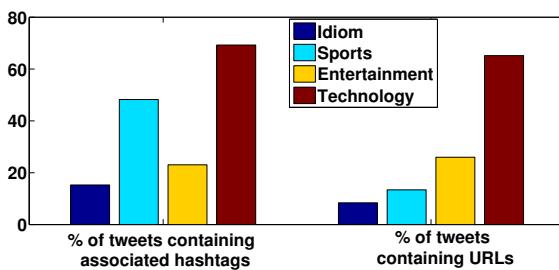


Fig. 24.1. Comparing topical trends and idioms: Percentage of tweets which contain (i) other hashtags (apart from the trend under consideration), and (ii) URLs. All values averaged over all trends of a particular category.

more likely to be accompanied by other hashtags and URLs related to the corresponding event in the off-line world. For instance, the sports-related trend #LIVvCHE (referring a match between the two English soccer clubs Liverpool and Chelsea) is accompanied by the hashtag #Torres which indicates a player who is a part of the match. On the other hand, Twitter-specific idioms are very seldom accompanied by other hashtags since they are not related to external websites or news-stories in the off-line world.

We also observed the timeline evolution of trends, i.e., how they start getting tweeted and become popular in Twitter. Expectedly, most topical trends emerge as a result of some related event in the off-line world, such as a sports or musical event, or a socio-political incident / issue. In case of idioms, an interesting pattern observed is that many idioms *initially* propagate along with hashtags related to some specific event in the off-line world. For example, the idiom ‘#MyFavouriteActor’ first appeared with the hashtag ‘#PeoplesChoice’ (related to the People’s Choice awards), while the idiom ‘#SexRequirements’ initially appeared with the health-related hashtag ‘#FitnessPromo’. These idioms, however, follow their independent path with users innovating interesting comments and thus making them popular.

We also attempted to analyze the spread of idioms across different geographical regions. Note that it is difficult to directly infer the geographical region from which a tweet is posted, since very few tweets are geo-tagged, and only a small fraction of Twitter users specify their geographical location in the profile [18]. Hence, we use the *language* of the user posting a tweet (which is mentioned in the profile for most of the users) and *language* of the tweet (which is present in the profile of each tweet) itself as a proxy for that tweet’s geographical location.

For each trend, we note the number of distinct languages mentioned in the profiles of all the tweets containing that trend, and all the users who posted a tweet containing that trend. Figure 24.2 shows, for the idioms, cumulative distributions (CDF) for the number of distinct languages in which the trends are discussed. We see that in general, idioms are discussed over several languages. Most of the idioms draw participation from users from many diverse locations and languages. This is interesting as most of the idioms are English language words but it goes beyond the langauge and penetrates fast. In the figure it is shown that the diversity of user’s language is more than the language in which the tweet is written — this happens because many *non English-speaking* users tweet in English.

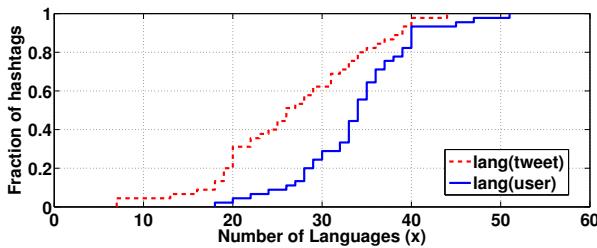


Fig. 24.2. Distribution of the number of distinct languages in which the idioms are discussed.

24.5.2. Characterizing users interested in various categories

In order to understand the nature of the users who are interested in promoting particular types of trends, we identify sets of users who are interested in the different categories (sports / entertainment / technology / idioms), and compare various characteristics of these users.

24.5.2.1. Identifying users interested in a certain category:

To identify users who are interested in a certain category, we identify those users who *frequently* discuss trends of that category. For a particular category, we initially consider all the users who have posted at least one tweet on a trend in that category. We rank the users based on the number of different trends in that category on which they have posted at least one tweet. Subsequently, for each category, we consider the top 10,000 users according to this ranking, i.e., the 10,000 users who have posted most number of distinct trends in that category (according to our dataset) are considered.

Since our objective is to identify users who are genuinely interested in trends of a certain category, we next attempt to verify whether the users selected above frequently discuss trends on that category. For this, we collected the 3,200 most recent tweets for each of the selected users, by crawling their time-line through the Twitter API, and used our classifier *Odin* to classify the hashtags contained in these tweets, to check what fraction of these hashtags were related to that category. For instance, for a certain user u included among the top 10,000 users who posted on most sports-related trends in our dataset, we checked whether a significant fraction of all hashtags included in u 's recent tweets were related to sports. Additionally, Opencalais (www.opencalais.com) tool is used to identify the topic of each tweet present in the timeline of a user. We included a user

in the final selected set for a category, if at least 30% of the hashtags and 70% of the tweets posted by her (among her recent tweets) were judged to be related to that category.

In this way, we finally identified a set of 5,000 users who are genuinely interested in each of the four categories. We refer to these sets of users as *idiom-users*, *sports-users*, *entertainment-users*, and *technology-users*. The rest of this section studies the characteristics of these sets of users.

24.5.2.2. Popularity of the users

We start by checking the popularity of the users interested in the various categories. We use two standard metrics of popularity of users in the Twitter social network [19, 20] – (i) follower-count, i.e., the number of followers of a given user, and (ii) listed-count, i.e., the number of Twitter Lists a given user is included in. Both metrics resulted in very similar observations. Figure 24.3 and Figure 24.4 show the distribution of the listed-count and follower-count values of users who predominantly discussed the trends in the four categories.

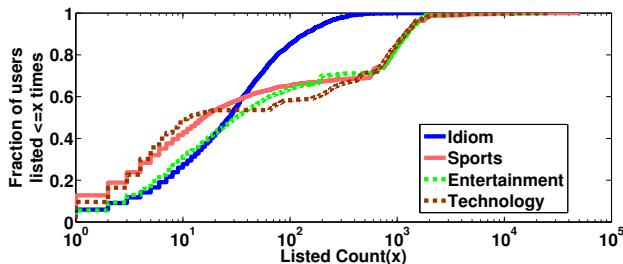


Fig. 24.3. CDF of listed count of users of various categories.

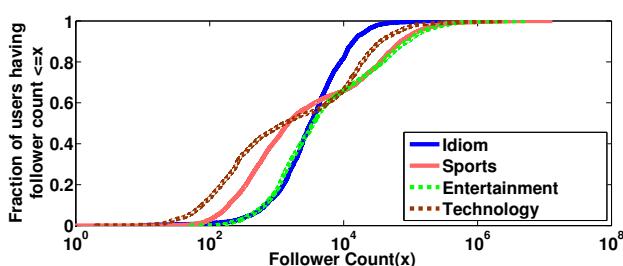


Fig. 24.4. CDF of follower count of users of various categories.

We observe an interesting trend. Almost *all* idiom-users are relatively less popular – 65% of the idiom-users have listed-count values in the range 0–40. In contrast, a significant fraction of the users who predominantly discuss the topical trends(sports, entertainment, technology) are very popular users, which includes experts from their respective fields. The above statistics lead to some interesting insights. There seems to be two very distinct types of users who dominantly participate in discussions on topics related to the off-line world (e.g., sports, entertainment, technology) – (i) popular users who are experts on these topics (e.g., researchers, sports-persons, journalists, musicians), and (ii) the common masses who are interested in these topics. This agrees with findings in recent research studies [3, 21]. In sharp contrast, users who dominantly participates in idioms are mostly common masses.

24.5.3. Characterizing the users

We start by attempting to characterize the users, i.e., identify who they are and what topics they are interested in.

Inferring characteristics of the users: To infer the characteristics of a given user u , we refer to two sources – (i) the bio of u , which is a short autobiography written by a user to describe herself, and (ii) the name and description of Twitter Lists in which u is included as a member. While the bio says how u describes herself, the List-names say *how other users describe u* – List-names and descriptions have been used by prior studies [22, 23] to infer the topical attributes of the users who are members of the Lists.[‡] For a given category, we consider the bio (or Lists) of all the 5000 users chosen for this category (as described above), and find the words which occur in the bio (or Lists) of most number of these users. The bio and List-names are pre-processed using standard techniques such as case-folding, removal of a common set of stopwords, and so on.

[‡]Lists in Twitter are a feature by which a user can group together accounts on a topic of her interest. For instance, a user u can create a List named ‘Music’ and add the accounts of famous musicians and singers such as LadyGaga, AshtonKutcher to the List. It is important to note that if user u creates a List on ‘Music’ and adds user v to the List, then this List serves two purposes – first, it tells us that the user-account v is likely to be an expert on the topic music [22, 23], and second, it indicates that the user u is interested in music.

Table 24.5 shows the top 5 words which appear in the bio and Lists of the users for each category. As expected, the users for the topical categories are characterized by words related to the topics. For instance, the sports-users are described using words such as ‘wrestling’, ‘wwe’, ‘players’, names of teams such as ‘dodgers’ and ‘lakers’, and so on. Similarly, the technology-users are described by words such as ‘iphone’, ‘android’, ‘macbook’, and so on. On the other hand, the idiom-users are mostly described by words related to day-to-day conversation and positive sentimental words such as ‘love’, ‘life’, ‘faves’, and so on.

Table 24.5. Characterizing the users who frequently discuss trends in each category – top 5 words appearing in (i) the user-bio, and (ii) Lists in which the users are members.

Idiom		Sports		Entertainment		Technology	
Bio	Lists	Bio	Lists	Bio	Lists	Bio	Lists
bands	faves	sports	wwe	5sos	band	news	social
life	ily	game	wrestling	justin	album	tech	media
girls	luke	fan	sports	bands	music	phone	tech
love	nigg	wrestling	chelsea	ariana	youtubers	oracle	tweet
cool	styles	football	fans	luke	idols	software	business

Inferring topics of interest of the users: We next explore the topics of interest of the three categories of users. For this, we check (i) the most frequent words in the tweets posted by the users, and (ii) the most frequent words in the names and descriptions of the Lists created by these users. Table 24.6 shows the 5 most frequent words in the tweets posted by the users, and the names of Lists created by these users. We again find that sports-users are primarily interested in sports-teams and sports-events, and the entertainment-users mostly discuss about ongoing musical concerts and actors (e.g., ‘5sos’, ‘harry’). On the other hand, the idiom-users are primarily interested in daily conversations and chatting.

Table 24.6. Inferring the interests of users who frequently discuss trends in each category – top 5 words appearing in (i) the tweets posted by the users, and (ii) Lists which are created by these users.

Idiom		Sports		Entertainment		Technology	
Tweet words	List names	Tweet words	List names	Tweet words	List names	Tweet words	List names
harry	girls	sports	wwe	show	band	android	news
faves	ily	game	sports	music	video	ios	tech
boys	friend	tna	football	5sos	music	phone	software
love	life	john	lakers	photo	album	google	technology
school	follow	nba	clubs	video	artist	apple	company

It is evident that the sports-users and technology-users have similar interest as what is expected – they are primarily interested in sports and technology respectively. However, the idiom-users require a better characterization; for this, we asked three volunteers to independently observe their profiles, their tweeting patterns, and the Lists created by the idiom-users. All the volunteers had similar observations about the idiom-users, which are summarized below.

- The idiom-users are mostly interested in gossiping with each other about their day-to-day activities, in the offshoots of entertainment-related topics such as ongoing ‘5sostaughtus’, ‘whyharrydidthat’ etc, and in phrases related to celebrities and music events. As an evidence, Figure 24.5 and Figure 24.6 show box-plots for the *average number of hashtags and URLs per tweet* posted by the users of the four groups. It is evident that the idiom-users post much fewer hashtags and URLs as compared to the sports-users, entertainment-users, and technology-users. The presence of hashtags and URLs indicates information about specific topics in a tweet [24, 25]; hence, their absence in the tweets posted by the

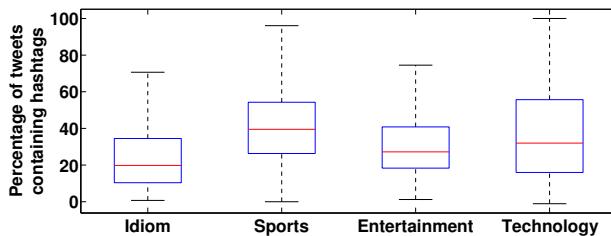


Fig. 24.5. Average number of hashtags per tweet posted by the four groups of users.

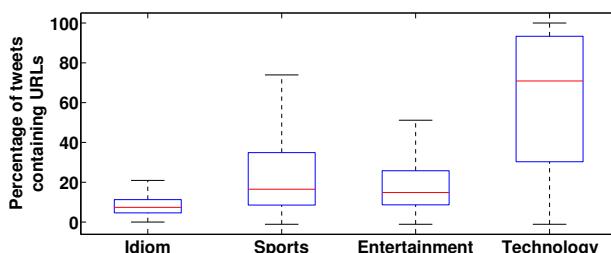


Fig. 24.6. Average number of URLs per tweet posted by the four groups of users.

idiom-users shows that these users seldom post about any specific events or topics that are discussed in the external Web or the off-line world.

- Manual observation of the Lists *created* by the idiom-users shows that more than 90% of these Lists are personal groupings (rather than topical), e.g., school friends, friends who meet at common parties, and so on. Hence, these users appear to be using Twitter mostly to stay in touch with their friends in the off-line world, rather than to acquire information about specific topics. In other words, these users are primarily using Twitter as a ‘social network’ rather than a ‘social media’ [1].

24.5.4. *Studying the interactions among the users*

We now investigate how the users in the four groups (i.e., groups of users who primarily post about politics, sports, entertainment and idioms) interact among themselves. In Twitter, the primary ways by which a user u can interact with another user v are (i) u can subscribe to the content posted by v by following v , or by following a List which has v as its member, and (ii) u can @mention v in her tweet.

24.5.4.1. *Analyzing interaction networks*

We construct two types of interaction networks among the users. The first is a *subscription network* where a directed link $u \rightarrow v$ indicates that user (node) u subscribes to the content posted by user v . The second is a *mention network* where the link $u \rightarrow v$ indicates that user u has @mentioned v .

To quantify the level of interaction among the users, we measure two structural properties of the subscription and mention network – (i) *density*, which measures what fraction of all links which can be present in a network, are actually present, and (ii) *reciprocity*, which indicates what fraction of the directed links are reciprocated, i.e., both the links $u \rightarrow v$ and $v \rightarrow u$ exist in the network. The importance of reciprocity is that if two users share a reciprocal link, then the two users are *mutual friends* with a higher probability (as compared to the chance of a fan subscribing to a celebrity, but the celebrity not reciprocating).

Table 24.7 shows the reciprocity and density of the mention and subscription networks among different groups of users. We find that the density of the subscription network among the idiom-users is significantly higher compared to that for the sports-users, entertainment-users, and technology-

Table 24.7. Reciprocity and density of the mention and subscription networks among different groups of users.

User-group	Mention Network		Subscription Network	
	Reciprocity	Density	Reciprocity	Density
Idiom	21.88%	0.0012	49.57%	0.0221
Sports	14.67%	0.0017	10.19%	0.0030
Entertainment	13.40%	0.0010	13.76%	0.0058
Technology	13.91%	0.0011	4.87%	0.0025

users. Also, the reciprocity is significantly higher for both the subscription network and the mention network for idiom-users, indicating that a large fraction of the interactions are between mutual friends. These observations indicate that, just like users interested in a common topic (sports, entertainment or technology), the idiom-users form their own group; in fact, they subscribe to / mention one another much more frequently than the topical groups of users.

Note that the density of the mention networks are comparable for all the user-groups. This is because, as observed earlier, the sports-users, technology-users, and entertainment-users contain a large number of common (less popular) users and a few popular celebrities, and most of the @mentions result from the common users mentioning the celebrities. For instance, a significant fraction of the @mentions among technology-users are directed towards @twitter and a few software companies. However, the reciprocities are lower for the topical groups, since the celebrities do *not* mention the common users. On the other hand, most of the mentions among the idiom-users (who have similar popularity) come from conversations among mutual friends, leading to high reciprocity. In fact, as much as 62.5% of the mentions among the idiom-users are between two users who share a *reciprocal* link in the corresponding subscription network (i.e., are likely to be mutual friends), whereas this percentage is less than 35% for the topical user-groups.

24.5.4.2. Nature of conversations among the users

Finally, we analyze the nature of the conversations among the users of the same group. Specifically, when a user retweets or mentions another user *in the same group*, we check whether the hashtags used in the tweets are related to the common topic of interest of the users. For instance, among the hashtags which a sports-user retweets or mentions to another sports-user along with the tweets, we check what fraction of such hashtags are

related to sports. For this, we use our classifier *Odin* to classify hashtags present in the tweets where a user mentions or retweets another user from the same-group. The results are shown in Table 24.8. More than 74% of the hashtags that are mentioned / retweeted among the sports-users, entertainment-users, and technology-users are related to the corresponding common topic of interest of that user-group. In sharp contrast, only about 25% of the hashtags that are exchanged among idiom-users are idioms. This again shows that idiom-users are not a focused topical group rather they engage themselves in diverse issues.

Table 24.8. Percentage of hashtags (present in tweets) where a user of a certain group mentions or retweets another user of the same group, which are related to the topic of interest of that user-group.

User-group	Idioms	Sports	Entertainment	Technology
% of topical hashtags in retweeted tweets	22.83%	78.47%	81.63%	79.57%
% of topical hashtags in mentioned tweets	25.74%	74.58%	77.12%	78.54%

24.5.5. Type of user-groups and their identifiability

Our analyses reveal that the group of users interested in Twitter-specific idioms has very different characteristics compared to the groups of users interested in topics such as technology, sports and entertainment. In this section, we attempt to explain the differences and their implications on identifiability of the groups.

24.5.5.1. Explaining group formation

Formation of user-groups in a social network has been a long-standing topic of research in sociology, and several theories have been proposed to explain their formation [10, 11]. According to the well-accepted *common identity and common bond theory* [12, 13, 15], there are two primary types of groups. In *identity-based groups*, people join the group due to their interest in a well-defined common theme (topic), whereas *bond-based groups* are driven by personal social relations (bonds) among the members, and may be characterized by the absence of any common topic of discussion. As a result, bond-based groups have higher reciprocity among the members than identity-based groups. Also, the discussions in bond-based groups tend to vary widely and cover multiple subjects, while in identity-based groups, they tend to be related to the common topic of interest of the group.

The above analyses on the four user-groups show that, as expected, the users interested in a common topic like sports, entertainment or technology form identity-based groups, with fewer interactions (@mentions) among friends, and most of the discussions among the members being related to the topic of common interest (Table 24.8). On the other hand, the idiom-users group is characterized by relatively higher levels of personal interactions with mutual friends, and a relatively small fraction of the conversation among the friends is related to their common topic i.e. idioms. Hence, the idiom-users form a bond-based social community within Twitter, in which they discuss their personal topics of interest as well as conversational matters.

24.5.5.2. Identifiability of the groups

The differences in the nature of various user-groups can have significant impact on the identifiability of the groups. To demonstrate this, we used two algorithms for detecting groups in the Twitter social network, and checked how well they could identify the idiom-users group and the topical groups.

(i) We used the well-known Infomap community detection algorithm [6] on the Twitter subscription network among all the users spanning the four user-groups. Then we enumerated the number of different communities identified by Infomap, where the members in any of the four user-groups are distributed. Table 24.9 (second row) shows that the topical groups were scattered into significantly higher number of Infomap communities, as compared to the idiom-users group.

Table 24.9. (i) Number of communities identified by Infomap, into which a user-group is scattered, (ii) average number of topical groups assigned per user by the topical group identification approach developed in [3].

User-group	Idioms	Sports	Entertainment	Technology
Number of communities	107	284	272	281
Number of groups assigned per user	9	2	2	3

(ii) Bhattacharya *et al.* [3] proposed a methodology to identify *topical communities* in Twitter (comprising of users who are experts on a topic or interested in the topic). We used this method to check the number of distinct topical communities a member in our dataset is placed. We found that, on average, a user in the idiom-users group is placed in many more topical communities, than a user in the sports-users / entertainment-users

and technology-users groups (Table 24.9, last row).

These observations reveal that within Twitter, there exist two different kinds of network structure – one is an information network, and other one is social communication network. Any community detection method which considers only one facet of the network might not be able to identify all the communities accurately.

24.6. Conclusion

The popular perception of the research community is that, there are two parts of Twitter – one interesting part where participants read and post a wide variety of topical tweets, and another part which comprises of pointless babble and is hence unimportant and uninteresting. However, in our study, we find that these pointless babbles, even though not related to any off-line event, frequently become trending in Twitter due to participation of large number of common masses. These users form bond-based groups among themselves to discuss their personal interests – idioms and some other forms of fun and gossip. This study has several implications, such as for community detection in social networks. Keeping in mind the popularity of idioms, we feel that the Twitter platform should provide a service where idiom lovers can easily find trending idioms and users who post idioms actively and frequently.

24.7. Acknowledgments

K. Rudra was supported by a fellowship from Tata Consultancy Services, and A. Chakraborty was supported by Google India PhD Fellowship and the Prime Minister Fellowship for Doctoral Research. S. Ghosh was supported by a postdoctoral fellowship from the Alexander von Humboldt Foundation.

References

- [1] H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, pp. 591–600 (2010).
- [2] S. A. Myers, A. Sharma, P. Gupta, and J. Lin. Information network or social network? the structure of the twitter follow graph. In *Proceedings of the 23rd International Conference on World Wide Web Companion*, pp. 493–498 (2014).

- [3] P. Bhattacharya, S. Ghosh, J. Kulshrestha, M. Mondal, M. B. Zafar, N. Ganguly, and K. P. Gummadi. Deep twitter diving: Exploring topical groups in microblogs at scale. In *Proceedings of the 17th ACM Conference on Computer-Supported Cooperative Work and Social Computing*, pp. 197–210 (2014).
- [4] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web*, pp. 851–860 (2010).
- [5] D. M. Romero, B. Meeder, and J. Kleinberg. Differences in the mechanics of information diffusion across topics: Idioms, political hashtags, and complex contagion on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, pp. 695–704 (2011).
- [6] M. Rosvall and C. T. Bergstrom, Maps of random walks on complex networks reveal community structure, *Proceedings of the National Academy of Sciences*. **105**(4), 1118–1123 (2008).
- [7] K. Lee, D. Palsetia, R. Narayanan, M. M. A. Patwary, A. Agrawal, and A. Choudhary. Twitter trending topic classification. In *Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops*, pp. 251–258 (2011).
- [8] A. Zubiaga, D. Spina, R. Martínez, and V. Fresno, Real-time classification of twitter trends, *Journal of the Association for Information Science and Technology*. **66**(3), 462–473 (2015).
- [9] M. Naaman, H. Becker, and L. Gravano, Hip and trendy: Characterizing emerging trends on twitter, *Journal of the American Society for Information Science and Technology*. **62**(5), 902–918 (2011).
- [10] D. McMillan and D. Chavis, Sense of community: A definition and theory, *Journal of Community Psychology*. **14**(1), 6–23 (1986).
- [11] E. Wenger, *Communities of practice: Learning, meaning, and identity*. Cambridge University Press, New York, NY, USA (1999).
- [12] D. A. Prentice, D. T. Miller, and J. R. Lightdale, Asymmetries in attachments to groups and to their members: Distinguishing between common-identity and common-bond groups, *Personality and Social Psychology Bulletin*. **20**(5), 484–493 (1994).
- [13] Y. Ren, R. Kraut, and S. Kiesler, Applying common identity and bond theory to design of online communities, *Organization Studies*. **28**(3), 377–408 (2007).
- [14] K. Sassenberg, Common bond and common identity groups on the internet: Attachment and normative behavior in on-topic and off-topic chats, *Group Dynamics Theory Research And Practice*. **6**(1), 27–37 (2002).
- [15] P. A. Grabowicz, L. M. Aiello, V. M. Eguiluz, and A. Jaimes. Distinguishing topical and social groups based on common identity and bond theory. In *Proceedings of the 6th ACM International Conference on Web Search and Data Mining*, pp. 627–636 (2013).
- [16] G. Berardi, A. Esuli, D. Marcheggiani, and F. Sebastiani. Isti@trec microblog track 2011: Exploring the use of hashtag segmentation and text quality ranking. In *Proceedings of the 20th Text Retrieval Conference* (2011).

- [17] G. D. Forney, The viterbi algorithm, *Proceedings of the IEEE*. **61**(3), 268–278 (1973).
- [18] B. Hecht, L. Hong, B. Suh, and E. H. Chi. Tweets from justin bieber's heart: The dynamics of the location field in user profiles. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 237–246 (2011).
- [19] M. Cha, H. Haddadi, F. Benevenuto, and K. P. Gummadi. Measuring user influence in twitter: The million follower fallacy. In *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media (ICWSM)*, pp. 10–17 (2010).
- [20] S. Ghosh, N. Sharma, F. Benevenuto, N. Ganguly, and K. P. Gummadi. Cognos: Crowdsourcing search for topic experts in microblogs. In *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 575–590 (2012).
- [21] S. Wu, J. M. Hofman, W. A. Mason, and D. J. Watts. Who says what to whom on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, pp. 705–714 (2011).
- [22] N. Sharma, S. Ghosh, F. Benevenuto, N. Ganguly, and K. P. Gummadi. Inferring who-is-who in the twitter social network. In *Proceedings of the 2012 ACM Workshop on Workshop on Online Social Networks*, pp. 55–60 (2012).
- [23] C. Wagner, V. Liao, P. Pirolli, L. Nelson, and M. Strohmaier. It's not in their tweets: Modeling topical expertise of twitter users. In *Proceedings of the 3rd IEEE international Conference on Social Computing/Privacy, Security, Risk and Trust*, pp. 91–100 (2012).
- [24] J. Teevan, D. Ramage, and M. R. Morris. #twittersearch: A comparison of microblog search and web search. In *Proceedings of the 4th ACM International Conference on Web Search and Data Mining*, pp. 35–44 (2011).
- [25] S. Ghosh, M. B. Zafar, P. Bhattacharya, N. Sharma, N. Ganguly, and K. P. Gummadi. On sampling the wisdom of crowds: Random vs. expert sampling of the twitter stream. In *Proceedings of the 22nd ACM International Conference on Conference on Information and Knowledge Management*, pp. 1739–1744 (2013).

Chapter 25

Sampling Theorems for Twitter: Ideas from Large Deviation Theory

Deepan Palguna^{1,*}, Vikas Joshi^{2,†}, Venkatesan Chakravarthy^{2,‡},
Ravi Kothari^{2,**} and L. V. Subramaniam^{2,††}

¹*School of Electrical and Computer Engineering,
Purdue University, USA*

²*IBM Research India, New Delhi, India*

*dpalguna@gmail.com, †vijoshij@in.ibm.com, ‡vechakra@in.ibm.com,
**rkothari@in.ibm.com, ††vsubram@in.ibm.com

The daily volume of Tweets generated by Twitter is around 500 million, and the impact of this data on applications ranging from public safety, opinion mining, news broadcast, etc., is increasing day by day. Analyzing large volumes of Tweets for various applications would require techniques that scale well with the number of Tweets. In this chapter we discuss a theoretical formulation for sampling Twitter data. Metrics to quantify the statistical representativeness of the Tweet samples are introduced, and results on the number of samples sufficient to obtain highly representative Tweet samples are derived. These statistical metrics quantify the representativeness or goodness of the sample in terms of restoring public sentiments associated with these frequent keywords. Sampling a sufficient number of Tweets uniformly and randomly could serve as a first step before using other sophisticated summarization methods to generate summaries for human use. Experiments conducted on real Twitter data are provided as examples to show how the bounds behave in practice. Moreover, we compare different kinds of random sampling algorithms in these experiments. The bounds derived are attractive since they do not depend on the total number of Tweets in the universe. Although these ideas and techniques are specific to Twitter, they could find applications in other areas as well.

25.1. Introduction

The number of Twitter users and Tweets have steadily increased with about 500 Million Tweets being generated daily. Twitter data is interesting and

important since the expressions of sentiment on various events and topics [1] encapsulate rich knowledge that may be used in many applications.

Twitter provides a host of APIs to crawl Tweets based on user preferences. Access to 100% of the real-time Tweets (with filtering capability) is provided by Twitter's Powertrack API. Although several options are available to access real-time Tweets, past Tweets can be accessed only using Historical Powertrack. Processing 100% of the Tweets can result in accurate analysis of user behavior, while it requires a significant amount of computational and storage footprints. Not surprisingly, to make the data size more easily manageable, there are other APIs that provide access to a subset of the Tweets. Twitter's free streaming API allow users to access Tweets in real-time, and can filter Tweets based on three parameters: keywords, user IDs and geographical bounding boxes. However, the streaming API provides at most (randomly sampled) 1% of the total Tweets generated. The 1% limitation can be overcome by using Twitter Decahose that provides access to 10% of the total Tweets.

The subsets obtained through the Streaming API or through filtering based on a specified criteria makes the computation much more tractable in situations with limited access to computational resources. However, one is also prompted to wonder if there is a (highly informative) subset of the overall Tweet population. Potentially, such a highly informative subset would reflect the characteristics of the overall tweet population.

Topic modeling in Twitter [2] is typically done using distributions over keywords which could include nouns like the names of personalities, places, events, etc. Twitter is also widely used to convey opinions or sentiments about these nouns. Motivated by this, we consider two statistical properties to quantify the representativeness of a sample. These are the frequency of occurrence of nouns and the conditional frequencies of sentiments, conditioned on each frequent noun. These frequencies constitute the conditional sentiment distribution of the noun. In order to compute this distribution, we assume that there are three possible sentiments for each Tweet—positive, negative or neutral. The conditional frequencies are computed as empirical frequencies as explained in later sections. In the analysis and experiments presented in this chapter, we assume that a perfect parts of speech tagger [3] provides the information on whether a word is a noun or not. Similarly, we assume that our sentiment analysis algorithm accurately classifies the sentiment of each Tweet into one of the three categories.

An overview of the results discussed in this chapter is presented now. We theoretically derive sufficient conditions on the number of samples needed

so that:

- a) the conditional sentiment distributions of frequent nouns in the universe are close to the corresponding conditional sentiment distributions in the sample.
- b) the dominant sentiment conditioned on a noun is also dominant in the sample.

The exact statements of these results will be made in later sections. The bounds derived in this chapter are attractive since they do not depend on the number of Tweets in the universe, and depend only on the maximum length of a Tweet which can be readily bounded. Theoretical bounds are evaluated by performing experiments with real Twitter data, which makes the bounds interesting to data providers and researchers in order to decide how much data to use in their respective applications. They could be used as a simple first step in extractive summarization or opinion mining in order to save on computation, especially when Tweets spanning multiple days are to be analyzed.

25.2. Related Work

Summarization algorithms for Twitter aim to extract representative Tweets from a large universe for human consumption. These algorithms are closely related to document summarization methods [4]. A comparison of various methods for summarizing Twitter data can be found in [5]. Existing Tweet summarization methods are often computationally expensive and would not scale to millions of Tweets, which is typical of the daily Tweet volume. Therefore, as mentioned previously, random sampling could serve as a first step before using any of these algorithms for human readable summaries, provided the random sample is representative of the universe.

There have been many papers that deal with the quality of samples given by Twitter API's free 1% sample. These papers are primarily empirical in nature. In [6] the authors compare the API's feed with the Tweets obtained from Firehose—which is also provided by Twitter, but contains all the Tweets instead of a sample. In [7], the authors empirically compare random sampling against sampling done with the help of human experts. In [8], the authors analyze the bias in Twitter's API without using the costly Firehose data. The effects of using multiple streaming APIs is considered by [9]. As can be seen, there have been many papers on the empirical analysis of the quality of Tweets summaries and the Twitter API itself. It is also understood that random sampling preserves statistical properties of the

universe. But an analytical treatment of the problem from the viewpoint of deriving conditions on sample sizes needed to produce representative samples is a relatively unexplored area and the results in this chapter are among the first ones to theoretically characterize the behavior of random Tweet sampling.

25.3. Tweet Sampling Problem Formulation

The set of Tweets that forms the universe of Tweets is denoted by T . This set contains N Tweets, each of which can have a maximum of L words. From N Tweets, we uniformly and randomly sample K Tweets with replacement. The set of Tweets in the sample is denoted using S . Each Tweet is modelled as a bag of words. Moreover, we assume that each Tweet has an associated sentiment, which can be in one of three possible classes—positive, negative or neutral. Some notations used throughout the chapter:

$$K_w = \#\text{Tweets in } T \text{ that contain noun } w$$

$$K_w^S = \#\text{Tweets in } S \text{ that contain noun } w$$

$$K_{w,p} = \#\text{positive sentiment Tweets in } T \text{ that contain noun } w$$

$$K_{w,p}^S = \#\text{positive sentiment Tweets in } S \text{ that contain noun } w$$

$$f_w = \frac{K_w}{N}, \quad f_w^S = \frac{K_w^S}{K} \text{ (Noun frequency in universe & sample)}$$

$$f_{w,p} = \frac{K_{w,p}}{K_w}, \quad f_{w,p}^S = \frac{K_{w,p}^S}{K_w^S} \text{ (Conditional sentiment frequency)}$$

Similar notations are used for minus (negative) and neutral sentiments, where we replace p (plus) in the subscript with m and n to denote minus and neutral respectively. The ordered triple of $(f_{w,p}, f_{w,m}, f_{w,n})$ forms the sentiment distribution conditional on noun w . As mentioned in the introduction, we derive sufficient conditions on the number of samples needed so that important statistical properties of the universe are retained in the sample with the desired probability.

Definition 25.1. For a number $\theta \in [0, 1]$, a noun whose frequency in T is at least θ is said to be a θ —frequent noun.

Lemma 25.1. *The number of nouns that occur with frequency at least θ is upper bounded by L/θ .*

Proof. Total number of words $\leq NL$. A θ -frequent noun would consume at least $N\theta$ words. So number of θ -frequent nouns cannot exceed L/θ . \square

Results derived in large deviation theory come in handy while deriving sufficient sample size bounds for statistically representative Tweet samples. In this respect one of the most useful bounds is the Chernoff bound which bounds the tail probability of the sum of independent random variables. There are a few different ways in which the bound can be stated but we use one of the statements for the binomial random variables in this chapter.

Theorem 25.1 (Chernoff Bound). *If random variable V is a binomial random variable obtained as the sum of independent and identically distributed Bernoulli random variables, then for any $0 < \mu < 1$, we have*

$$\begin{aligned}\mathbb{P}\{V \leq (1 - \mu)\mathbb{E}[V]\} &\leq e^{-\frac{\mu^2 \mathbb{E}[V]}{2}} \text{ and} \\ \mathbb{P}\{V \leq (1 + \mu)\mathbb{E}[V]\} &\leq e^{-\frac{\mu^2 \mathbb{E}[V]}{3}}.\end{aligned}\quad (25.1)$$

In this chapter, we deal with empirical frequencies that can be viewed as the ratio of the sum of Bernoulli random variables and a fixed number, resulting in a situation where such bounds can be applied with good effect.

We now clarify two points related to our theoretical analysis. Firstly, the sampling algorithm we use for analysis is a *batch* sampling method, which fixes a sample size and then does uniform random sampling with replacement. But when Twitter's API gives a 1% sample, it is very likely that it would perform *sequential* sampling, where each Tweet in the universe would be sampled one-by-one and independent of all other Tweets with probability 1%. We use batch sampling for analysis since the analysis is much cleaner than the case of sequential sampling. Experiments with real data suggest that the performance of both methods depend strongly only on the number of samples, and not on whether sampling is done in batch mode or in sequential mode.

Secondly, our sampling method is random and it may theoretically be possible to derive equations for the exact probabilities of the associated events as a function of K . But computing them and solving the resulting non-linear equations for the sufficient K would be very difficult. Doing this might give stronger bounds for K , but its infeasibility motivates us to derive sufficient conditions on K using bounds on the probabilities instead of the exact probabilities. The sufficient sample sizes would depend only on parameters that govern probabilistic goodness guarantees demanded by

the application and not on N , which is an added benefit. In the next two sections, we describe the statistical properties that we want to retain, the associated probabilistic goodness guarantees, and state our results on the sample size bounds.

25.4. Sampling for Sentiment Restoration

Since Twitter is used as an important medium for voicing opinions, it is important for the conditional sentiment distribution of nouns in the summary to be close to the corresponding distribution in the universe.

Definition 25.2. For some $\lambda \in [0, 1]$, we say that the sentiment distribution of word w is *not restored* if the following event occurs

$$\{|f_{w,p}^S - f_{w,p}| > \lambda\} \cup \{|f_{w,m}^S - f_{w,m}| > \lambda\} \cup \{|f_{w,n}^S - f_{w,n}| > \lambda\} \quad (25.2)$$

We say that a failure occurs if even a single θ -frequent noun's sentiment distribution is not restored.

We want to sample enough Tweets so that the probability of failure is bounded above by $h \in [0, 1]$.

Theorem 25.2. If $K \geq \frac{\log((h\theta)/(6L))}{\log(1 - \theta + \theta \exp\{-\lambda^2/3\})}$, then the probability of failure to restore sentiment is less than h .

Proof. The event of the sentiment distribution of word w not being preserved is:

$$\{|f_{w,p}^S - f_{w,p}| > \lambda\} \cup \{|f_{w,m}^S - f_{w,m}| > \lambda\} \cup \{|f_{w,n}^S - f_{w,n}| > \lambda\}$$

First consider the event $\{|f_{w,p}^S - f_{w,p}| > \lambda\}$ conditional on K_w^S . Conditional on a realization of K_w^S , $K_{w,p}^S$ is a binomial random variable with parameters K_w^S and $f_{w,p}$. So we can use Chernoff's bound [10] for binomial random variables to derive an upper bound on the probability of the event.

$$\begin{aligned} \mathbb{P}\left\{|f_w^S - f_{w,p}| > \lambda \middle| K_w^S\right\} &= \mathbb{P}\left\{K_{w,p}^S > K_w^S \frac{K_{w,p}}{K_w} \left(1 + \lambda \frac{K_w}{K_{w,p}}\right) \middle| K_w^S\right\} \\ &\quad + \mathbb{P}\left\{K_{w,p}^S < K_w^S \frac{K_{w,p}}{K_w} \left(1 - \lambda \frac{K_w}{K_{w,p}}\right) \middle| K_w^S\right\} \\ &\leq \exp\left(-\frac{\lambda^2 K_w^S}{3f_{w,p}}\right) + \exp\left(-\frac{\lambda^2 K_w^S}{2f_{w,p}}\right) \\ &\leq 2 \exp\left(-\frac{\lambda^2 K_w^S}{3f_{w,p}}\right) \end{aligned} \quad (25.3)$$

We can remove the conditioning by taking expectation over all possible realizations of K_w^S to get:

$$\mathbb{P}\{|f_{w,p}^S - f_{w,p}| > \lambda\} \leq \mathbb{E}\left[2 \exp\left(-\frac{\lambda^2 K_w^S}{3f_{w,p}}\right)\right] \quad (25.4)$$

The expectation in the Eq. 25.4 is over the realizations of K_w^S . This expectation is the moment generating function of a binomial random variable with parameters (K, f_w) evaluated at $-\lambda^2/(3f_{w,p})$ [11]. Using the union bound and adding the bounds for the three sentiments we have

$$\begin{aligned} & \mathbb{P}\{\text{Sentiment not preserved for noun } w\} \\ & \leq 6[1 - f_w + f_w \exp(-\lambda^2/(3 \max\{f_{w,p}, f_{w,m}, f_{w,n}\}))]^K \\ & \leq 6[1 - f_w + f_w \exp(-\lambda^2/3)]^K \end{aligned} \quad (25.5)$$

In order to bound the probability of failure, which is the probability that the sentiment distribution of atleast one θ -frequent noun is not preserved, we again apply the union bound over all θ -frequent nouns. The function of f_w on the R.H.S. of Eq. 25.5 decreases with f_w . Since we have taken w to be θ -frequent, this expression is maximized by substituting θ in place of f_w . Moreover, the bound on the number of θ -frequent nouns is L/θ , which gives the upper bound on the probability of failure to preserve sentiment for any θ frequent noun as $(6L/\theta)[1 - \theta + \theta \exp(-\lambda^2/3)]^K$. So if $K \geq \frac{\log((h\theta)/(6L))}{\log(1 - \theta + \theta \exp\{-\lambda^2/3\})}$ then the probability of failure will not exceed h —proving Theorem 25.2. \square

In opinion mining, one may not want to guarantee that the entire distribution conditioned on noun w be restored. One may rather be interested to simply guarantee that the dominant sentiment in the universe also dominates in the sample.

Definition 25.3. A sentiment is said to *dominate* if its frequency conditioned on w is higher than the other two conditional frequencies.

As an example of a dominant sentiment, if noun w is such that $f_{w,p} > f_{w,m}$ and $f_{w,p} > f_{w,n}$, then positive sentiment is said to be the dominant sentiment in the universe. The goal of dominant sentiment preservation is to sample enough Tweets so that this property is *preserved* with high probability in the sample as well i.e., $f_{w,p}^S > f_{w,m}^S$ and $f_{w,p}^S > f_{w,n}^S$. For deriving the sufficient number of Tweets to guarantee this, we develop a preliminary result in Lemma 25.3. Let (X_1, X_2, X_3) be jointly multinomial

random variables with parameters (M, f_1, f_2, f_3) , where $f_1 + f_2 + f_3 = 1$ and $f_1 > f_2 > f_3$. Consider first

$$\begin{aligned}\mathbb{P}\{X_3 > X_1\} &= \mathbb{E}[\mathbb{P}\{X_3 > X_1 \mid X_2 = x_2\}] \\ &= \mathbb{E}[\mathbb{P}\{X_3 > (M - x_2)/2 \mid X_2 = x_2\}]\end{aligned}\quad (25.6)$$

Lemma 25.2. *Conditional on $X_2 = x_2$, X_3 is binomial with parameters $(M - x_2, f'_3 = f_3/(f_1 + f_3))$.*

Proof. To prove this, we derive the conditional distribution and show that it is of the required form.

$$\begin{aligned}\mathbb{P}\{X_3 = x_3 \mid X_2 = x_2\} &= \frac{\mathbb{P}\{X_3 = x_3, X_2 = x_2\}}{\mathbb{P}\{X_2 = x_2\}} \\ &= \frac{\mathbb{P}\{X_3 = x_3, X_2 = x_2, X_1 = M - x_2 - x_3\}}{\mathbb{P}\{X_2 = x_2\}} \\ &= \frac{\binom{M}{M-x_2-x_3, x_2, x_3} f_1^{M-x_2-x_3} f_2^{x_2} f_3^{x_3}}{\binom{M}{x_2} f_2^{x_2} (1-f_2)^{M-x_2}} \\ &= \binom{M-x_2}{x_3} (f'_3)^{x_3} (1-f'_3)^{M-x_2-x_3}\end{aligned}\quad (25.7)$$

This proves Lemma 25.2. \square

Lemma 25.3. *For $\delta_3 = \frac{1}{2f'_3} - 1$,*

$$\mathbb{P}\{(X_3 > X_1) \cup (X_2 > X_1)\} \leq 2 \left[f_2 + (1-f_2) \exp\{-(\delta_3^2 f'_3)/(2+\delta_3)\} \right]^M$$

Proof.

$$\begin{aligned}\mathbb{P}\{X_3 > (M - x_2)/2 \mid X_2 = x_2\} &= \mathbb{P}\left\{X_3 > (M - x_2)f'_3\left(\frac{1}{2f'_3} - 1 + 1\right) \mid X_2 = x_2\right\} \\ &\leq \exp\left\{-(\delta_3^2(M - x_2)f'_3)/(2 + \delta_3)\right\}\end{aligned}\quad (25.8)$$

The application of Chernoff's bound to get Eq. 25.8 is possible since $f_1 > f_3$ resulting in $f'_3 < 1/2$. The marginal distribution of $M - X_2$ is binomial with parameters $(M, 1 - f_2)$. Taking expectation w.r.t. X_2 , we get

$$\mathbb{P}\{X_3 > X_1\} \leq [f_2 + (1-f_2) \exp\{-(\delta_3^2 f'_3)/(2+\delta_3)\}]^M$$

By symmetry

$$\mathbb{P}\{X_2 > X_1\} \leq [f_3 + (1-f_3) \exp\{-(\delta_2^2 f'_2)/(2+\delta_2)\}]^M$$

Now we can use the union bound and write the probability that X_1 is smaller than either X_2 or X_3 as

$$\begin{aligned}\mathbb{P}\{(X_3 > X_1) \cup (X_2 > X_1)\} &\leq [f_3 + (1 - f_3) \exp\{-(\delta_2^2 f'_2)/(2 + \delta_2)\}]^M \\ &\quad + [f_2 + (1 - f_2) \exp\{-(\delta_3^2 f'_3)/(2 + \delta_3)\}]^M\end{aligned}\tag{25.9}$$

The bound in Eq. 25.9 can be further simplified by considering $\frac{\delta_3^2}{2+\delta_3} f'_3$. This is equal to $\frac{f'_3}{(2f'_3-1)(4f'_3+1)}$, which is a decreasing function of f'_3 if $f'_3 < 1/2$. This holds because we have taken f'_3 to be $\frac{f_3}{f_1+f_3}$ and assumed that $f_1 > f_3$. Since $f_2 > f_3$, we have $f'_2 = \frac{f_2}{f_1+f_2} > \frac{f_3}{f_1+f_3} = f'_3$. Consequently, $\exp\{-\frac{\delta_3^2}{2+\delta_3} f'_3\} > \exp\{-\frac{\delta_2^2}{2+\delta_2} f'_2\}$. Now we can write the bound in Eq. 25.9 as

$$\mathbb{P}\{(X_3 > X_1) \cup (X_2 > X_1)\} \leq 2[f_2 + (1 - f_2) \exp\{-(\delta_3^2 f'_3)/(2 + \delta_3)\}]^M\tag{25.10}$$

This concludes the proof of Lemma 25.3. \square

We can use Lemma 25.3 with similar notations to derive a bound on the number of Tweets that are adequate to *preserve* the dominant sentiment for noun w . We relabel $f_{w,p}$, $f_{w,m}$ and $f_{w,n}$ using $f_{w,1}$, $f_{w,2}$ and $f_{w,3}$ such that they are ordered as $f_{w,1} > f_{w,2} > f_{w,3}$. The same notation is followed for $K_{w,1}$, $K_{w,2}$ and $K_{w,3}$. Similarly, we use $f'_{w,3} = \frac{f_{w,3}}{f_{w,1}+f_{w,3}}$ and $\delta_{w,3} = \frac{1}{2f'_{w,3}} - 1$. Based on Lemmas 25.2 and 25.3, we can now state a result on preserving the dominant sentiment of a noun.

Theorem 25.3. *For $h \in [0, 1]$, the probability that noun w 's dominant sentiment is not dominant in the sample is upper bounded by h if*

$$K \geq \frac{\log(h/2)}{\log\left(1 - f_w + f_w(f_{w,2} + (1 - f_{w,2}) \exp\left\{-\frac{\delta_{w,3}^2}{2+\delta_{w,3}} f'_{w,3}\right\})\right)}$$

Proof. We follow a similar proof by conditioning first on a realization of K_w . Conditioned on K_w , $(K_{w,1}, K_{w,2}, K_{w,3})$ are jointly multinomial with parameters $(K_w, f_{w,1}, f_{w,2}, f_{w,3})$. From Lemma 25.3, we have

$$\begin{aligned}\mathbb{P}\{\text{Noun } w\text{'s sentiment not preserved} \mid K_w\} \\ \leq 2[f_{w,2} + (1 - f_{w,2}) \exp\{-(\delta_{w,3}^2 f'_{w,3})/(2 + \delta_{w,3})\}]^{K_w}.\end{aligned}\tag{25.11}$$

We average over K_w again whose marginal distribution is binomial with parameters (K, f_w) . This gives the probability generating function [11].

$$\begin{aligned} & \mathbb{P}\{\text{Noun } w\text{'s sentiment not preserved}\} \\ & \leq 2 \left[1 - f_{w,2} + f_w(f_{w,2} + (1 - f_{w,2}) \exp \left\{ - \frac{\delta_{w,3}^2}{2 + \delta_{w,3}} f'_{w,3} \right\}) \right]^K. \end{aligned}$$

So if

$$K \geq \frac{\log(h/2)}{\log \left(1 - f_w + f_w(f_{w,2} + (1 - f_{w,2}) \exp \left\{ - \frac{\delta_{w,3}^2}{2 + \delta_{w,3}} f'_{w,3} \right\}) \right)}.$$

then the probability of noun w 's sentiment is not preserved is lesser than h . This proves Theorem 25.3. \square

25.5. Simulations and Experiments

25.5.1. Data set description

Using Twitter's free API, we created a data-set based on certain important political events of 2014, which we call the PE-2014 data-set. This is done by using a comprehensive vocabulary to filter and retain only those Tweets that pertain to certain political events. These Tweets form our Tweet universe with a volume of $\sim 200,000$ Tweets per day. Although this is substantially smaller than the total daily Tweet volume, it could still be too large a set for more sophisticated summarization algorithms, especially if we want to summarize Tweets from multiple days. Our simulation results are obtained over one day's worth of data (March 22, 2014).

25.5.2. Theoretical bounds

To characterize the bounds given by our theoretical results, we need an estimate or an upper bound for L . A loose upper bound for L would be 140, which is the maximum number of characters allowed per Tweet. For these results though, we estimate L from our PE-2014 data-set to be 33 words. Figure 25.1 shows the bound in Theorem 25.2 as a function of the probability of failure to restore the sentiment distribution of all θ -frequent nouns. This is plotted for different values of the deviation λ and θ . The leftmost panel of Fig. 25.2 compares the theoretical bounds for dominant sentiment preservation (Theorem 25.3) and sentiment distribution restoration for a single noun. For the same failure probability, the bounds on the sample size for preserving the dominant sentiment are much smaller than

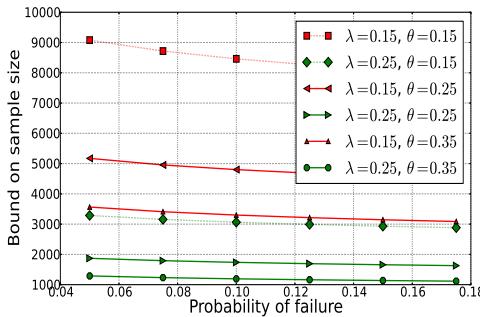


Fig. 25.1. Bounds for sentiment restoration over all θ -frequent nouns as a function of failure probability.

the bound on the sample size for restoring the entire distribution. In the middle panel of Fig. 25.2, we compare the two cases of dominant sentiment preservation shown in the left panel. One would expect the case with $f_{w,1} = 0.9, f_{w,2} = 0.06, f_{w,3} = 0.04$ to have a smaller sample size bound as compared to $f_{w,1} = 0.45, f_{w,2} = 0.35, f_{w,3} = 0.2$, due to the closeness of frequencies in the latter case. This intuition is confirmed in middle panel. It is also possible to encounter cases when the bounds for order preservation are larger than the bounds for distribution restoration as seen in the rightmost panel of Fig. 25.2. Since the conditional frequencies of the sentiments are very close to each other ($f_{w,1} = 0.35, f_{w,2} = 0.33, f_{w,3} = 0.32$), the number of samples needed to preserve their order exceeds the number of samples needed to restore their distribution for these parameters. In such cases though, preserving sentiment order may not be important since human users may conclude that no sentiment really dominates.

25.5.3. Experiments with Twitter data

We measure the sample quality in terms of statistical properties and failure events related to the theoretical formulation, and compare them with the theoretical results for a range of sample sizes. We measure the maximum deviation in sentiment for word w as

$$\max\{|f_{w,p}^S - f_{w,p}|, |f_{w,m}^S - f_{w,m}|, |f_{w,n}^S - f_{w,n}|\} \quad (25.12)$$

The maximum sentiment deviation for θ -frequent nouns measured over 100 Monte Carlo rounds is shown in the second column of Table 25.1.

We see from this column that it is enough to sample 2000 Tweets to achieve a mean maximum sentiment deviation of 0.0952 for $\theta = 0.1$. These

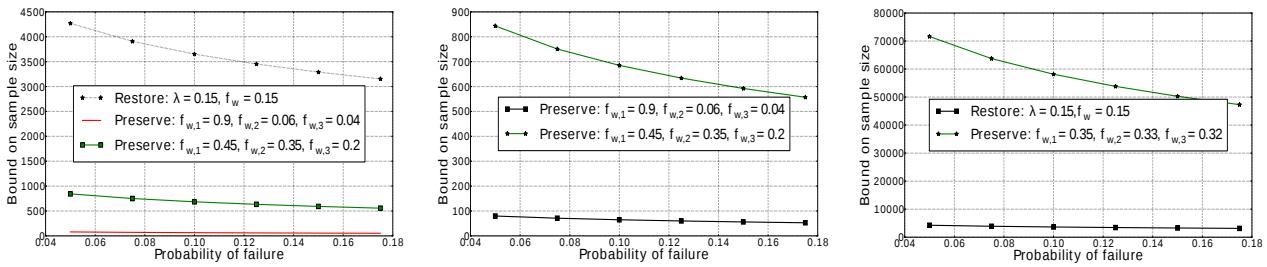


Fig. 25.2. Sentiment distribution restoration vs. dominance preservation for one noun with $f_w = 0.15$, allowed deviation $\lambda = 0.15$.

Table 25.1. Mean maximum deviation over all θ -frequent nouns in PE-2014, shown for $\theta = 0.1$ and 100 Monte Carlo rounds.

# samples (K)	Mean max. deviation
2000	0.0952
4000	0.0706
8000	0.0488
10000	0.0426

agree with the sufficient conditions in Theorem 25.2 and are quite close to the bounds shown in Fig. 25.1. Table 25.2 shows that the number of errors in sentiment order preservation for three most frequent nouns is negligible. The last line is for a fourth noun whose sentiment order is intuitively hard to preserve.

Table 25.2. Error events in preserving the dominant sentiment for PE-2014 data, shown for three most frequent nouns and 100 Monte Carlo rounds. The fourth noun is an illustration of a case with difficult order preservation. Two different sample sizes and their corresponding error counts are shown as ordered tuples for each noun.

Noun number	Noun frequency (f_w)	Sentiments ($(f_{w,p}, f_{w,m}, f_{w,n})$)	Number of samples	Errors in order restoration
Noun 1	0.25	(0.31, 0.25, 0.44)	(2000, 4000)	(1, 0)
Noun 2	0.14	(0.32, 0.19, 0.49)	(2000, 4000)	(0, 0)
Noun 3	0.12	(0.23, 0.27, 0.50)	(2000, 4000)	(0, 0)
Noun 4	0.054	(0.378, 0.255, 0.367)	(2000, 4000)	(39, 38)

25.5.4. Comparison with sequential sampling

In sequential sampling we go through the Tweets in the universe one-by-one and sample a Tweet independent of others with probability p . For the same data set, we perform sequential sampling with two different p values—0.01 and 0.02. For each value of p , we perform 100 Monte Carlo rounds and measure the same metrics as in batch sampling. These results are shown in Table 25.3. From this table, we see that the performance of sequential sampling and random sampling with replacement are very similar, and are primarily influenced by the sample size alone.

25.6. Conclusion

In this chapter we have derived bounds on the number of random samples needed to guarantee statistically representative Tweet samples. Random

Table 25.3. Error events in PE-2014 data shown for $\theta = 0.1$ and 100 Monte Carlo rounds. The number of sentiment order preservation errors in the last column is the sum over the 3 most frequent nouns shown in Table 25.2.

p	[Avg. # of samples]	Mean max. deviation	# sentiment order preservation errors
0.01	2385	0.0883	0
0.02	4790	0.0627	0

sampling can be used as a first step before using other sophisticated algorithms to form human readable summaries or for mining social opinion. For example, if we want to find out Twitter sentiment for a keyword that we know is frequent in the universe, then we could run our sentiment analysis tools on a much smaller random sample whose size does not depend on N . Therefore, using random sampling as a first step could provide significant computational savings for many such applications with practical accuracy requirements. Our experiments with real data confirm that the bounds agree with the sample sizes needed in practice. Our results would readily apply to restoring topic models represented as mixture distributions in order to sample a diverse set of Tweets. These ideas also easily extend to applications beyond Twitter to other short messages such as call center transcripts.

References

- [1] Y. Mejova, P. Srinivasan, and B. Boynton. Gop primary season on twitter: "popular" political sentiment in social media. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pp. 517–526, ACM, New York, NY, USA (2013). ISBN 978-1-4503-1869-3. doi: 10.1145/2433396.2433463. URL <http://doi.acm.org/10.1145/2433396.2433463>.
- [2] L. Hong and B. D. Davison. Empirical study of topic modeling in twitter. In *Proceedings of the First Workshop on Social Media Analytics*, pp. 80–88 (2010).
- [3] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python*, 1st edn. O'Reilly Media, Inc. (2009). ISBN 0596516495, 9780596516499.
- [4] A. Nenkova and K. McKeown. A survey of text summarization techniques. In *Mining Text Data*, pp. 43–76, Springer US (2012). ISBN 978-1-4614-3222-7. URL http://dx.doi.org/10.1007/978-1-4614-3223-4_3.
- [5] D. Inouye and J. K. Kalita. Comparing twitter summarization algorithms for multiple post summaries. In *Privacy, security, risk and trust (passat), 2011 ieee third international conference on and 2011 ieee third international conference on social computing (socialcom)*, pp. 298–306 (2011).
- [6] F. Morstatter, J. Pfeffer, H. Liu, and K. M. Carley, Is the sample good

- enough? comparing data from Twitter's streaming API with Twitter's Firehose, *Proceedings of ICWSM* (2013).
- [7] S. Ghosh, M. B. Zafar, P. Bhattacharya, N. Sharma, N. Ganguly, and K. Gummadi. On sampling the wisdom of crowds: Random vs. expert sampling of the twitter stream. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pp. 1739–1744 (2013).
 - [8] F. Morstatter, J. Pfeffer, and H. Liu. When is it biased?: Assessing the representativeness of twitter's streaming api. In *Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion*, pp. 555–556 (2014).
 - [9] K. Joseph, P. M. Landwehr, and K. M. Carley. Two 1% s dont make a whole: Comparing simultaneous samples from twitters streaming api. In *Social Computing, Behavioral-Cultural Modeling and Prediction*, pp. 75–83. Springer (2014).
 - [10] H. Chernoff, A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations, *The Annals of Mathematical Statistics*. **23**(4), 493–507 (12, 1952). URL <http://dx.doi.org/10.1214/aoms/1177729330>.
 - [11] A. Papoulis and S. Pillai, *Probability, random variables, and stochastic processes*. McGraw-Hill electrical and electronic engineering series, McGraw-Hill (2002).

Chapter 26

A Machine-mind Architecture and Z*-numbers for Real-world Comprehension

By Romi Banerjee and Sankar K. Pal

*Center for Soft Computing Research
Indian Statistical Institute, Kolkata, India
rm.banerjee@gmail.com, sankar@isical.ac.in*

This article documents our efforts on machine-mind design for real-world, understanding. It includes: a) enumeration of essential macro-architectural elements and working principle of our machine-mind framework, b) definition of a novel paradigm, the Z^* -numbers, for encapsulation of essential objective and subjective elements of real-world information, and c) a detailed example illustrating the dynamics of the framework and the role of the Z^* -numbers. Our work is based on Minsky's theories of cognition, Zadeh's Z-number philosophy, and human brain processes of comprehension. The ideas herein, envision man-machine symbiosis and aim to contribute to the synthesis of an empathetic artificial mind. Among others, the primary novelty in our design lies in the emphasis on representation of the machine-self and its derivates towards emulation of bespoke real-world comprehension.

26.1. Introduction

The human brain is a continually evolving, self-organizing computing system that acquires, constructs, stores and processes real-world data, to find structure in them and make sense of the world around. An artificial cognitive ‘thinking’ system inspired by the human brain and aiming to work synergistically with human beings, therefore, must emulate each of these properties. Such systems need to analyze problems in keeping with the context, identify objectives, consider multiple solution strategies and activate scheme(s) that will lead the system to an acceptable goal state in reasonable time.

Cognitive or ‘thinking’ machines were pioneered by Turing [1]. This article elucidates our work towards the design and synthesis of such a ma-

chine for ‘man-machine symbiosis’ [2]. We describe here a framework of a sentient, social, computational mind, and a paradigm (the Z*-numbers) to encapsulate and allow computations on the objective and subjective aspects of real-world information.

Our design is founded on ‘non-symbolic theories’ [3] of cognition and mindfulness envisioned through Minsky’s ‘Society of Mind’ [4] and ‘Emotion Machine’ [5] models of the working mind, and dynamics [6, 7] of comprehension in the human-brain. Zadeh’s Z-numbers [8] have been extended to form the Z*-numbers.

Post a brief survey [9, 10] of existing models of cognition, we chose Minsky’s theories as the basis of our design because they appreciate the role of ‘fast and slow thinking’ [11], intuition, commonsense, affects, social-interactions, self-consciousness and rationalism in real-world comprehension. The indispensability of each of these elements in cognition is vouched for in developmental linguistics [6, 12], philosophy [13, 14] and neuroscience [15]. Sparse implementation initiatives of the ‘Emotion Machine’ further encouraged our endeavors.

“Could a social robot have a subjective world? I think that social robots will “require” some form of subjective world. It is possible to imagine a social robot without feelings, but that does not make it viable in practice. They might not be the same as yours or mine, but a social robot that has an episodic memory of the events in its past must have a first person experience that is unique to itself. Non-social robots (such as a self-driving car) don’t necessarily need one.” [16]

Coined in 2011, the Z-numbers [8] are an important contribution to the intelligent-systems design initiative. These constructs pioneer juxtaposition of the objective and subjective components of natural language expressions of the real-world towards enhanced comprehension of the same. This paradigm focuses on the precisiation of the *certainty* of information in a sentence.

Studies [15, 17–19] on the properties of the ‘self’ enumerate: a) a sense of ‘time’ to bind experiences into an unbroken narrative of one’s existence, b) the ‘context’ defining the subset of the environment relevant to a particular goal, and c) ‘emotions’, as some parameters that embody the sense of ‘self’. The Z*-numbers augment the Z-numbers with these elements towards improved summarization of natural language expressions. The machine-mind framework uses this paradigm to formulate tokens of internal thought processes.

The mechanisms of the machine-mind framework during comprehension and the role of the Z*-numbers therein have been illustrated through an example. The design has been analyzed through correspondence studies with the human brain and Minsky's theories. The article ends with a conceptual comparison between our framework and contemporary 'reflective sentient agent-design' initiatives. The novelty in our conceptualization lies in its emphasis on the notion of the sense of 'self' and its derivatives en route to bespoke comprehension of the real-world.

The design is currently in its very early stages and is subject to evolution with recurrent knowledge gain on the brain-processes.

The article is divided into the following sections: Section 26.2 presents an overview of the fundamental theories underlying our work; Section 26.3 describes our work; and Section 26.4 summarizes key elements of this article.

26.2. Fundamentals

This segment highlights theories inspiring our design – a synopsis of brain regions and functions thereof crucial to language processing (and comprehension in general), Minsky's model of cognition, and an outline of Zadeh's Z-numbers.

26.2.1. *The cerebral cortex in language processing*

The human brain, the most developed of all primate brains, serves as our reference guide in the design of a sentient, intelligent system.

The cerebral cortex is considered [7, 20, 21] to be the primary language processing center of the brain. It is divided into two almost symmetrical halves cerebral hemispheres, each made up of four lobes – connected by the corpus callosum. Table 26.1 and Table 26.2 list language processing functions of the lobes and memory categories of the brain, respectively. These serve as functional-requirements specifications for our system-design endeavours.

Our machine-mind is based on Minsky's 'Society of Mind' [4] and 'Emotion Machine' [5] theories, wherein he describes cognition through a 'layered-critic-selector-reflective' [31–34] framework.

Table 26.1. Cerebral cortex lobes and their roles in language comprehension.

Lobe	Lobe functions	Roles in language comprehension
Occipital	Processes visual signals and passes results to the parietal and temporal lobes.	Integrates visual information; relates current stimulus to past experiences and knowledge.
Frontal	Motor control; attention mediation; reasoning; judgment; decision making; problem solving; learning; strategic thinking; social behaviour; relates present to the future. Forms working-memory and prospective-memory [22].	Broca's area: syntax and morphology resolution; Self definition.
Parietal	Multimodal sensory information processing; spatial interpretation; attention mediation; language comprehension.	Angular gyrus: language and number processing, spatial cognition, memory retrieval, attention mediation, metaphor-comprehension [23, 24].
Temporal	Auditory perception; language comprehension and visual recognition; stores new semantic, episodic and autobiographical [25] memories; recognition-memory creation [26].	Wernicke's area: semantics resolution; Amygdala: affect processing, memory consolidation [27]; Hippocampus: short-term to long-term semantic and episodic memory consolidation, spatial navigation; Basal ganglia: reinforcement learning, procedural memory creation, priming and automatic behaviour, eye movement control [28], cognition [29].

26.2.2. Overview of the ‘Society of Mind’ and ‘Emotion Machine’

The ‘Society of Mind’ [4] construes the mind as an evolving, hierarchical, modular, computing mechanism. Components particularly relevant to our current design phase are:

Agent: Building blocks of the mind; represent an atomic component of a cognitive process.

Agency: Groups of *agents* realizing complex cognitive functions. The mind is a society of agencies.

K-lines: Data and control structures that turn on designated sets of agents. Types of K-lines:

Neme: Represents an idea (context) or a state of the mind.

Nome: Controls manipulation of representations and effects agencies.

Table 26.2. Categories of human memories.

Memory	Description
Working	Temporary memories of information on the task currently being handled.
Episodic	Detailed, predominantly contextual, representation of specific events; can last a life time; inspires emotions and personal associations with the event.
Semantic	Facts; relationships between facts; predominantly non-contextual; basis of abstractions of the real world.
Declarative	Ensemble of consciously stored and recalled episodic and semantic memories.
Procedural	Ensemble of memories pertaining to implicit learning; leads to automatic behaviour.
Long-term	‘Semantic codes’ [30] of declarative and procedural memories.
Short-term	‘Acoustic codes’ [30] of memories recalled for duration of the order of seconds without repetition.
Sensory	Memories of sensory stimuli, after it has ceased; is of the order of milliseconds.
Visual, Olfactory, Haptic, Taste, Auditory	Explicit memories of visual, olfactory, tactile or haptic, taste and auditory experiences, respectively.
Autobiographic	Subset of the episodic memory; deals exclusively with personal experiences.
Retrospective	Past memories, with respect to the current time-frame.
Prospective	Memories activated in the future, with respect to the current time-frame, by environmental cues; results in “remembering to remember” actions.

Frames [35]: Data structures depicting events and their constituent properties. These structures form hierarchical connected graphs of nodes and relations, where ‘top-level’ frames (or ‘frame-headers’) imply designated abstractions of an event, and terminal slots (or sub-frames) of ‘lower-level’ frames are instantiated to event-specific information.

Typical frame classes:

Surface Syntactic Frames: For verb and noun structures, prepositional and word-order indicator conventions.

Surface Semantic Frames: For action-centered meanings of words,

qualifiers and relations involving participants, instruments, trajectories and strategies, goals, consequences and side-effects.

Thematic Frames: For scenarios concerned with topics, activities, portraits, setting, outstanding problems and strategies commonly linked to a topic.

Narrative Frames: For scaffoldings of stories, explanations, arguments, and conventions about protagonists, plot forms and development, etc.; assists construction of new *thematic frame(s)*.

An ‘Emotion Machine’ [5, 34] is where the aforementioned components work across the ‘six levels’ of thinking illustrated in Figure 26.1. These involve recall, manipulation and organization of a vast repertoire of knowledge through powerful automated reasoning processes. Each of the layers incorporates ‘critics’ that consider the external world and internal system states, and activate ‘selectors’ to initiate ‘thinking’ on appropriate interpretation strategies for an event. Lower levels result in instinctive reactions to the external world, while the higher levels control these reactions in accordance with the system’s model of itself. These layers intuitively represent left brain (logico-deductive reasoning the four lower layers) and right brain

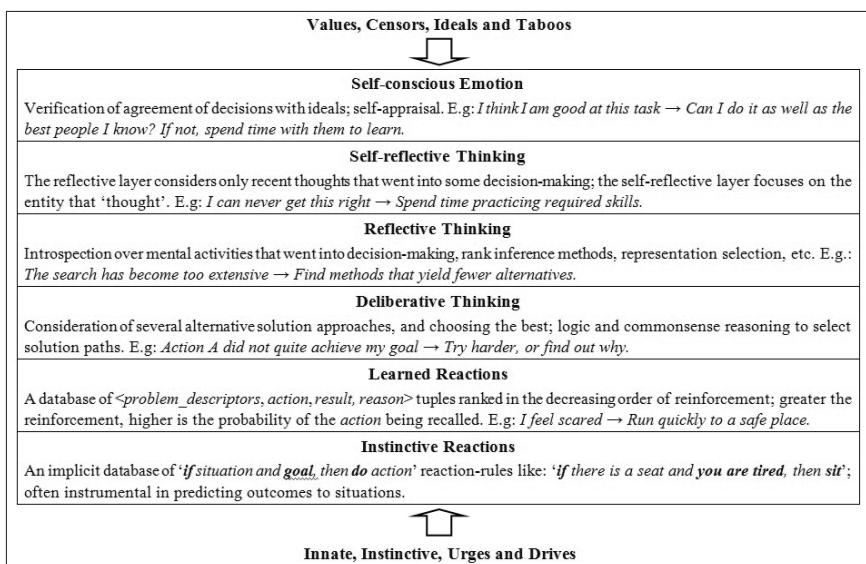


Fig. 26.1. Layers of cognition [5].

(subjective reasoning the top three layers) activities, and ‘slow and fast thinking’ [11]. The structure is reminiscent of Dennett’s ‘Tower of Intelligence’ [36] and the levels of cognitive-capacities [37] by Newen and Vogeley.

The Z^{**} -number paradigm, defined in Section 26.3.1.2., envisions a foundation for emulation of the two upper layers of Minsky’s model of the mind. It augments Zadeh’s Z-numbers [8] with essential parameters of subjectivity towards total encapsulation of real-world information.

26.2.3. Z-numbers

The Z-number [8] is a manifestation of ‘level-2’ [38] Computing With Words (CWW [39, 40]). It draws on concepts in [39–46] and pioneers attempts at precisiation of subjective perception(s) of a natural language statement by encoding the reliability or confidence in its constituent information. Consequently, if it were possible to simulate the endogenous arousal of beliefs on information, the Z-numbers would prove to be an effective means of representation of machine-subjectivity. Intuitively, it contributes to initiatives in man-machine symbiosis [2], Artificial General Intelligence (AGI) [47] and the ‘Intelligent Systems Revolution’ [1, 48].

Given a natural language statement, Y , the ‘Z-number’ of Y is a 2-tuple $Z = \langle A, B \rangle$, where A is the restriction (constraint) on the values of X (a real-valued uncertain variable implying the subject of Y) and B is a measure of the reliability (certainty) of A . Typically, A and B are expressed as words or clauses, and are both fuzzy numbers. Some examples of Z-numbers are:

- (i) $Y_1 = \text{This book is an absolute delight.}$

Therefore, $X = \text{Quality of the book,}$ and $Z = \langle \text{delightful, absolutely} \rangle.$

- (ii) $Y_2 = \text{It takes me about half an hour to reach point } A.$

Therefore, $X = \text{Time to reach point } A,$ and $Z = \langle \text{about half an hour, usually} \rangle.$

A is context-dependent and B summarizes the certainty or belief in the applicability of A , given $\langle X, \text{context of } Y_1 \rangle.$ B could either be explicitly worded or implied from Y (as in examples (i) and (ii), respectively).

The ordered 3-tuple $\langle X, A, B \rangle$ is referred to as a ‘Z-valuation’. A collection of Z-valuations is referred to as ‘Z-information’ and is the stimulus to a decision-making process.

Preliminary rules of Z-number based computations [8] are:

- (i) Precisiation of values of A and B through association with membership functions, μ_A, μ_B respectively.
- (ii) X and A together define a random event in R , and the probability of this event, p , may be expressed as:

$$p = \int_R \mu_A(u)p_X(u)d(u), \quad (26.1)$$

where u is a real-valued generic value of X and p_X is the underlying (hidden) probability density of X .

- (iii) The Z-valuation $\langle X, A, B \rangle$ is viewed as a generalized constraint on X , and is defined by:

$$\text{probability } (X \text{ is } A) \text{ is } B \\ \text{or, } p = \int_R \mu_A(u)p_X(u)d(u) \text{ is } B, \quad (26.2)$$

$$\text{i.e., } p = \mu_B \int_R \mu_A(u)p_X(u)d(u), \\ \text{subject to } \int_R p_X(u)d(u) = 1. \quad (26.3)$$

- (iv) Calculations using Z-numbers utilize the ‘Principle of Extension’. For example, consider the problem statement:

“It is probable that Mr. Smith is old. What is the probability that he is not?”

Let, X = Mr. Smith’s age, A = old, B = probable, C = not old, D = degree of certainty; $\mu_A, \mu_B, \mu_C, \mu_D$ are the membership functions associated with A, B, C and D respectively; p_X is the underlying (hidden) probability density of X ; and u is a real-valued generic value of X .

We therefore have: Probability $(X \text{ is } A)$ is B and need to evaluate Probability $(X \text{ is } C)$ is $?D$.

Using the Principle of Extension and expressions (26.2) and (26.3), we arrive at

$$\begin{aligned} \frac{\langle X, A, B \rangle}{\langle X, C, ?D \rangle} &= \frac{\text{probability } (X \text{ is } A) \text{ is } B}{\text{probability } (X \text{ is } C) \text{ is } ?D} \\ &= \frac{\mu_B \int_R \mu_A(u)p_X(u)d(u)}{\left(\int_R \mu_C(u)p_X(u)d(u) \right) \text{ is } ?D}, \end{aligned} \quad (26.4)$$

implying,

$$\mu_D(w) = \sup_{p_X} \left(\mu_B \int_R \mu_A(u)p_X(u)d(u) \right), \quad (26.5)$$

subject to $w = \int_R \mu_C(u)p_X(u)du$ and $\int_R p_X(u)d(u) = 1$.

Reference [49] presents a comprehensive discussion on Z-number based work. While most of these initiatives concentrate on the definition of Z-number calculi and its use in image processing, our [50, 51] focus is on its role in the extraction of the semantic-sense of natural language sentences. Major challenges in the implementation of the Z-numbers lie in the: a) representation of semantics of natural language elements in some form of machine-language, and b) emulation of endogenous arousal of certainty values and subsequent metacognition.

In this article, we describe our attempts to extend the philosophy of the Z-numbers – forming Z*-numbers – envisaging holistic representation of the information in a sentence, and in turn, the real-world. This is an effort in the direction of representation of the ‘machine-self’ and resolution of the aforementioned challenges as well.

The article is dedicated to a description of our work – enumeration of the components, principle and features of a machine-mind architecture for real-world cognition, definition of the Z*-number paradigm, an example demonstrating the dynamics of comprehension in the mind-framework and the role of the Z*-numbers therein, and analyses of our ideas.

26.3. Proposed Work

26.3.1. *The machine-mind framework*

Our machine-mind is founded on Minsky’s ‘Society of Mind’ [4] and ‘Emotion Machine’ [5] theories of the working mind. It also draws from questions, concepts and experimental evidence of human cognition across developmental psychology [12], psycholinguistics [6], philosophy of the mind, qualia and consciousness [13, 14], and neural underpinnings [15] of language and thought. Turing [1], Bush [52], McCarthy [53], Licklider [2] and key elements of language-comprehending systems [54–56] have influenced our design, as well.

References [57–59] describe popular realization attempts of the Society of Mind, and Refs. [60, 61] the Emotion Machine. The DeepQA architec-

ture [62], IDA [57], CMATTIE [59], LIDA [63], Genesis project [64] and Artificial Genie [65] are some in-progress contemporary initiatives of ‘conscious, sentient, reflective and learning’ agent-design. While some of these have demonstrated preliminary language-processing skills, all of them lack a sense of ‘self’ and ‘qualia’ and scientists wonder if they actually ‘understand’ [66].

Besides use of Minsky’s theories, the novelty in our work lies in the inclusion of the notion of the ‘machine-self’ – emulation of qualia, self-consciousness [53], commonsense reasoning, intuition, reflection and creativity to formulate bespoke granules of comprehension of the real-world. Though the framework constructs have been predominantly studied in their capacities as language-understanders, the principles contribute to real-world comprehension.

This article describes the macro-components – the ‘agencies’ and memory constructs – and working principle of our machine-mind architecture, followed by a study of its analogy with human brain processes of understanding. The micro-components, that is, agent-structures, system algorithms, and detailed memory data structure formats, are our current research concerns.

26.3.1.1. *Macro-components of the machine-mind architecture: Agencies and Memory Constructs [67]*

A. *Cognitive processes of comprehension*

Real-world cognition involves ‘iterative-incremental-developmental’ [6, 68] assembly of granules of comprehension of parts of an event (E) towards interpretation of the entire E . It entails:

Prediction – Causally relate the present to past experiences to envisage future actions; intuition, commonsense, learning and reflection play significant roles.

Visualization – Invoke real or intentional images of people, places, events, plots, etc.

Connection – Build factual or conceptual associations between intra-context components, inter-context events, and existing real-world knowledge and new information.

Question and **Clarification** – Reason and precisiate completeness, correctness and relevance of knowledge associations; rectify incorrect/incomplete concepts; reorganize ideas.

Evaluation – Test inter-perception coherence and prune insignificant or in-

correct concept-frames; attention-mediation; subjective or ‘self conscious’ assessments (emotions, degrees of interest, biases, etc.).

Intuitively, *prediction* and *visualization* involve all but the top two layers of Minsky’s mind-model (Figure 26.1); *connection* – the four lower layers; *question* and *clarification* – the learned, deliberative and reflective thinking layers; and *evaluation* – the top three layers.

Major cognitive processes constituting the above functions, in no specific order, are as follows. These processes are complex, mostly concurrent, and co-operative:

Symbol-extraction, Symbol-interpretation and Symbol-granulation – Extract and identify sensory inputs (or symbols) as digits, alphabets, special characters, etc.; group individual symbols into conceptual elements (words, numbers, phrases, clauses, sentences, etc.).

Syntax-resolution – Interpret grammatical sense of symbol-granules.

Semantic-resolution – Interpret context-sensitive sense of symbol-granules; involves –

Anaphora/cataphora-resolution – Resolve dependency (explicit, implicit or recursive) between objects and pronouns.

Spatio-temporal sense-resolution – Determine temporal and spatial senses of prepositional words or phrases.

Context-resolution – Isolate definitive factors of the setting for an event, statement or idea.

Sense-resolution – Activate appropriate context-sensitive sense of homonymous words or phrases; determine figure of speech.

Relevance-evaluation – Evaluate significance [69] of words/phrases processed; prune irrelevant concept-frames; attention-mediation.

Affect-evaluation – Monitor interest and affect progression; translate emotions into qualia [17, 18] and feelings [18].

Comprehension-evaluation – Test comprehension-correctness and completeness; initiate re-processing with respect to degree of comprehension and interest.

Frame-generation/retrieval/manipulation – Create, recollect and operate on frame-networks pertaining to the current concern; activate intuitive, commonsense, domain and real-world knowledge.

Encoding/Decoding – Translate frame-systems into compressed and indexed (by subjective elements) knowledge components and vice-versa; integrate multi-modal memories (visual, audio, tactile, etc.) of the same object/event.

Memory-handling – Retrieve declarative, procedural or sensory memories; consolidate information; monitor active frame status.

Error-handling – Disambiguate interpretations; resolve ‘cognitive biases’ [68, 70].

Instinctively, these functions engage multiple layers of the mind (*symbol-extraction/interpretation* – the two bottom layers, *symbol-granulation* – the learned reactions layer, others all the layers) and necessitate bi-directional information percolation. The information traversing bottom-up draws from sensorimotor inputs from the external (environment) or mental (imagination) space, while that traversing top-down relates to the individual’s sensibilities acquired through learning, experience and commonsense reasoning. *Frame-generation/retrieval/manipulation, encoding/decoding, memory-handling* and *error-handling* are baseline-functions that support all the processes preceding them in the above list.

B. Agencies and memory constructs

As is in a natural mind, the machine-mind typically processes simultaneously arriving real-world multi-modal sensory inputs, activates relevant experiences and knowledge across time, and iteratively executes functions listed in the preceding section to produce a network of hypergraphs of concept-associations for an event. Figure 26.2 is a black box depiction of the execution strategy of the framework, Figure 26.3 illustrates the machine-mind-agency-architecture, and Section 26.3.1.1.D summarizes its conceptual correspondence with Minsky’s theories and the cerebral cortex.

Based on the definition of ‘agencies’ for a Society of Mind model, we have categorized the mind-agencies of our framework into:

Super-agency – Denotes an operation-stack responsible for a complex cognitive functionality like ‘reasoning’ or ‘learning’.

Sub-agency – Super-agency functionality is achieved by a cluster of synchronized sub-agencies. Agents, simulating atomic sub-processes of cognition, constitute a sub-agency.

The super-agencies and constituent sub-agencies in a machine-mind are:

Sensory_Gateway (SG): Point of 2-way communication between the system and the external world; comprises ‘sensory’ sub-agencies [Primary: *Vision (V), Audition (A), Olfaction (O), Tactile (Tc), Taste (Ta); Secondary* [71]: *Balance (B), Temperature (Te), Pain (P) and Kinesthetic (K)*] to receive and activate processes on real-world inputs.

Deducer (De): The ‘brain’ of the system; processes **SG** signals to construct frames of comprehension. Comprises:

Syntax (Sy): Syntax-resolution; generates and manipulates surface syntactic frames.

Semantic (Se): Semantic-resolution; generates and processes surface semantic, narrative and thematic frames.

Self(Sf): Integrates **SG** faculties towards machine-‘proprioception’ [72] and manifestation of the sense [15] of ‘machine-self’; resolves cognitive biases; activates emotions and corresponding qualia [17, 18] and feelings [18].

Recall (Re): Maps entire problems or sub-problems to memories pertaining to the current and related contexts; handles knowledge reinforcements.

Creative (Cr): Hub of reflection, imagination, improvisation and ‘system IQ’ [43].

Summary (Su): ‘Censor’ [4] and ‘Suppressor’ [4] of the framework; consolidates memories; visualizes thoughts and evaluates degree of comprehension at strategic intervals; assists cognitive bias resolution.

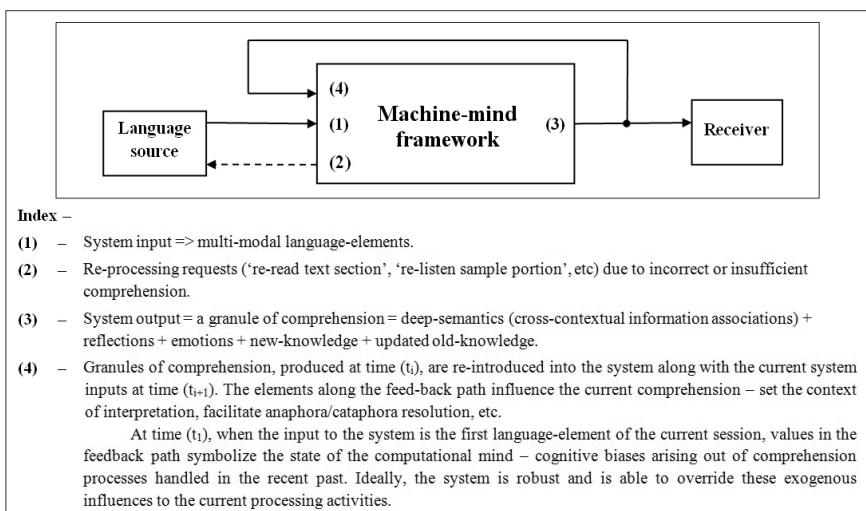


Fig. 26.2. Abstraction [67] of ‘iterative-incremental developmental’ comprehension strategy.

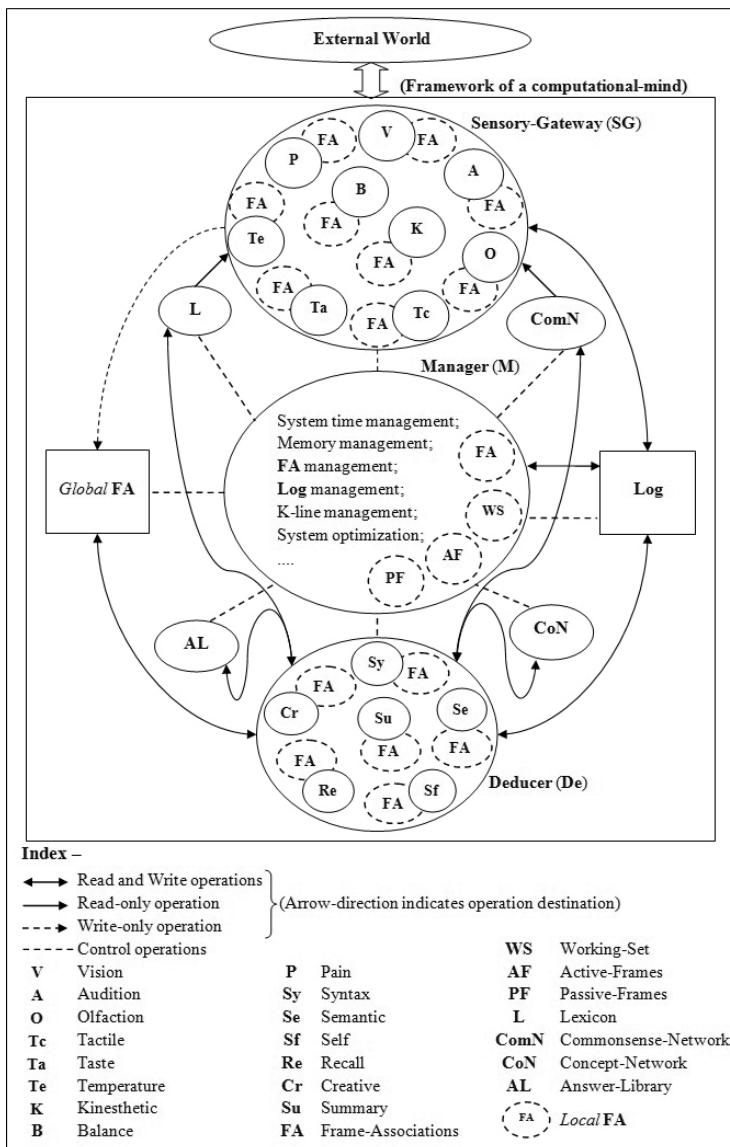


Fig. 26.3. The computational-mind framework [67].

Manager (M): System-administrator; runs in the background and is responsible for the activation and execution of ‘involuntary’ essential system functions (time management, memory handling, process synchronization, K-line management, frame encoding/decoding and compaction, system self-evaluation towards improvement, resource-arbitration, etc.).

Besides their individual functions, these agencies collaborate in the following ways to realize other crucial cognitive processes:

- (i) **Su, M:** Co-ordinate inter-agency activity; use heuristics and approximation schemes to prevent combinatorial explosions of thoughts and solution-strategies.
- (ii) **Se, Sf,** secondary **SG** sub-agencies (**B, K**): Create a sense of space and time.
- (iii) **Re, Cr, Su, Se, Sf:** Use effective indicators [73] of conscious thoughts and states of the system to arouse subjective interpretations of an event; resolve cognitive-biases; emulate self-motivation and innate mental-rewards.
- (iv) **Re, Cr, Su:** Constitute the ‘Difference-Engine’ [4].

Memory structures provide the basis [67] of cognition in all intelligent systems. The formats of these structures constrain: a) the nature and execution of system-algorithms and cognitive processes, b) storage and retrieval of experiences, and c) the rate of system-evolution and consequently system-stability. Drawing from these factors, the following dynamic memory constructs are an integral component of the framework:

Long-term knowledge-stores; conceptually resemble the memex [52]; draw inspiration from ConceptNet [74] and Hacker [54]:

Lexicon (L): System-vocabulary (words, phrases – facts, imagery, metaphors, etc.).

Answer-Library (AL): Repository of experiences(i.e., $\langle context_parameters, problem, solution_strategy, result, reasons \rangle$ tuples).

Concept-Network (CoN): Hypergraphs of frames of concepts – abstract and real – denoting domain and real-world knowledge; includes multi-modal sensory experiences.

Commonsense-Network (ComN): Hypergraphs of commonsense and intuitive or automatic behaviors; elements of **L**, **CoN** and **AL** are incorporated into **ComN** after prolonged reinforcement.

Basic *working-memory* structures; support deliberation and reflection by the system, and resemble ‘global workspaces’ [14, 75, 76] of blackboard [56]

architectures:

Log: Record of time-stamped agency-activities, system-states and partial-results, analyzing which – agencies are activated, error signals generated, backtracking and re-processing initiated, etc.; facilitates counter-factual thinking.

Frame-Associations (FA): Categorized into *global* and *local* (per sub-agency) types, where all recollected frames are placed in the former and relevant sections thereof transferred into the latter for deliberations by sub-agencies; **SG** sub-agencies use their *local FA* to store sensory memories; post problem-decomposition, **De** sub-agencies use their *local FA* to test multiple solution-perspectives of sub-problems before globally advocating (*<subproblem, solution, reason>* tuples) frame manipulation processes through **Log**; **Su** analyses these candidate sub-problem solution pools to extract combinations of effective full-problem handling strategies, and consequent frame-manipulations are reflected in *global FA*; **M** sub-agencies use their *local FA* to reason through system optimization mechanisms; sub-agencies can share sections or all of its *local FA* with other agencies.

The *system memory-management* constructs are used exclusively by **M** to support functionalities of the other agencies and system-‘mentalese’ [77]:

Working-Set (WS): Set of pointers to frame-networks, in **FA**, being referenced within a narrow time-window (of the order of seconds).

Active-Frames (AF): Set of pointers to frame-networks, in **FA**, being referenced within a broad time-window (of the order of minutes); **WS** is a subset of **FA**.

Passive-Frames (PF): Set of pointers to frame-networks, in **FA**, which were pruned out of **AF** due to irrelevance or lack of use. Instead of consolidating them into *long-term* memory, these frames remain available for the entire span of the current processing for quick ‘on-demand’ placement into **FA**.

Evidently, the framework describes a causal system (illustrated in Figures 26.2–26.4) and is based on distributed processing across the agencies. The data-driven design perspective reflects a shift away from the ‘word-at-a time’ [78] thinking underlying von-Neumann bottlenecks. Its modular structure allows local design-modifications. [See [67] for details on agency-functionalities, modes of operation, frame states and generic agency components of the framework. The design is prone to evolve as we learn more about the endogenous mind-mechanisms of cognition, with time.]

C. Working principle

Referring to functionalities of the components defined in the previous section, the framework operates as follows. Here we discuss text-understanding, but the principles apply to comprehension of real-world events as well (Figure 26.4 illustrates inter-agency activity during text processing):

Given a text-sample (T) to understand, V is endogenously activated and it begins extracting symbols from T . This triggers iterations of intuitive, commonsense, learned, deliberative, reflective and self-reflective actions by De – beginning with simple symbol granulation and interpretation, towards incremental processing of activated and newly formed knowledge and ideas (both real and abstract), and comprehension of T . The agencies operate on relevant **L**, **AL**, **ComN** and **CoN** memories and conceptualizations

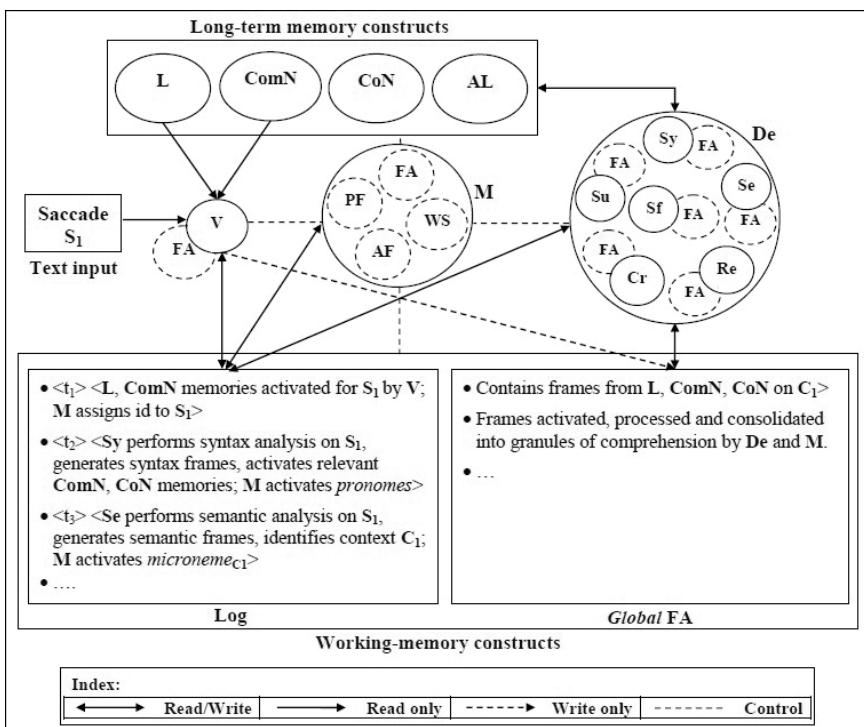


Fig. 26.4. Working principle [67] of the machine-mind architecture during text processing. [See [4] for notes on pronomes and micronemes.]

arising out of **De**'s operations. System activities are entered in **Log**, and memories and partial results are placed in *global FA* to facilitate agency-co-operation. Agency-operations involve encoding, decoding and integration of frame-types and data-types.

De continually assesses the status (familiarity, syntax, semantics, context, affects, interests, relevance, degree of comprehension, etc.) of the problem (words, clauses, sentences, paragraphs, memories, abstract ideas, visualizations, etc.) and opportunistically suggests interpretation-strategies and outcomes (predictions or absolute) en route to comprehension of *T*. These involve incremental-developmental iterations through - problem-decompositions, goal identification, information transfers across memory constructs; local and global frame-manipulation trials and executions; temporal trajectory adjustments; affective computing; broadcasts of potential success-rendering schemes and predicted results; signals to improvise or develop new ideas; alignment of interpretations with self-interests; information-consolidation; long-term memory updates and reinforcements, etc. – to realize the cognitive processes enumerated in Section 26.3.1.1.A.

V extracts text in saccades [6], the length, rate of extraction, and location of which is regulated by **De**, until reading and subsequent comprehension is complete. Intuitively, **V** and **De** work in tandem towards cognition of *T*, while **M** unobtrusively provides core system functionalities.

All activities are corroborated by corresponding **Log** entries. These entries are annotated with reasons facilitating operation-appraisal, consequent agency activation or inhibition, and reflection. Successful interpretation of *T* could result in new knowledge and updating or reinforcement of existing concepts.

An algorithmic analysis of the working principle necessitates detailed elucidation of frame-structures, time and space complexity analyses, and correctness and completeness verifications. As this article focuses on the top-level framework structures, we have provided hints on construct-parameters and tuples, but refrain from discussions on the micro-components.

D. Validation of completeness of machine-mind conceptualization

Considering that our architecture is based on Minsky's theories of cognition and human brain processes, functional coverage of these theories by our design would validate its completeness. Tables 26.3–26.5 illustrate these studies of correspondence.

Table 26.3. Coverage [67] of ‘layers of the mind’-functionality by the mind-agencies. [(*) in a cell indicates coverage of the layer-functionality (row) by the mind-agency (column).]

Mind layers \ Machine mind-agencies	V	Sy	Se	Sf	Re	Cr	Su	M
Mind layers								
Instinctive reactions: Accept sensory inputs; procedural memory updating	*							*
Learned reactions: Assign meaning to real-world stimuli symbol extraction, granulation; syntax and basic semantic analysis	*	*	*		*		*	*
Deliberative thinking: Context identification; sense-disambiguation; rhetoric and prosodic analysis; relevance and coherence analysis; concept-consolidation; comprehension- visualization; deliberative memory updating		*	*		*	*	*	*
Reflective thinking: Reason and optimize deliberations; curiosity and self-motivation arousal; build cross-contextual associations; counter-factual visualization; resolve cognitive biases; reform concept(s) identify new knowledge; clarify misconceptions		*	*		*	*	*	*
Self-reflective thinking: Monitor interest and comprehension progression; resolve cognitive biases; reform concept(s) identify new knowledge; clarify misconceptions; retrospective and prospective time-frame adjustment; self-motivation			*	*			*	*
Self-conscious emotions: Attachment of emotions or levels of interest and perceptions; awareness of social norms and ideals			*	*			*	*

Evidently, the enumerated machine-mind elements emulate behavior as per all the layers of the mind and the language-processing centers of the human brain. It is through the conceptual coverage of the reference structures that we consider our conceptualization complete. As has been highlighted earlier, the design process is prone to evolve with gain in knowledge on the seemingly innate mind-mechanisms of comprehension.

The next section describes our attempts to extend the philosophy of the Z-numbers to enhance its capacity of encapsulation of the semantics of an event – described in natural language or as a frame-network. We define here the augmented Z-number or Z*-numbers. These are envisioned to give expression to the ‘machine-self’ and form operands or tokens of system-‘mentalese’ [77] during cognition.

Table 26.4. Correspondence [67] between language-processing centers of the human brain and the machine-mind agencies.

Lobe	Parts / Functions of the lobes with respect to language cognition	Framework agencies
Occipital	Processes visual information	V
Frontal	<i>Broca's area</i> Self definition, attention, social behaviour Reasoning, judgment, strategic thinking	Sy Sf Re, Cr, Su
Parietal	<i>Angular gyrus</i>	Se, Sf, Su, Re, Cr
Temporal	<i>Wernicke's area</i> <i>Amygdala</i> <i>Hippocampus</i> <i>Basal Ganglia</i> Recognition	Se Sf, Su Su Sf, Su, Re Re

Table 26.5. Correspondence [67] between identifiable human memory categories the machine-mind memory constructs.

Human memory categories	Framework memory-constructs
<i>Working</i>	<i>global FA; local FA</i> of De and M sub-agencies; AF; PF
<i>Declarative</i>	CoN
<i>Procedural</i>	ComN
<i>Long-term</i>	L; CoN; ComN; AL
<i>Short-term</i>	First set of entries into <i>global FA</i> by SG
<i>Sensory</i>	<i>local FA</i> of SG sub-agencies
<i>Visual, Olfactory, Haptic</i>	Memories annotated by the senses they represent indicated by their data-types in ComN and CoN
<i>Taste, Auditory</i>	
<i>Autobiographic</i>	Subset of CoN
<i>Retrospective, Prospective</i>	Constructed out of ComN and CoN (PF supports the emulation of these memories)

26.3.1.2. *Z*-numbers: Towards representation of ‘machine-self’*

“Self-consciousness, i.e. the ability to observe some of one’s own mental processes, is essential for full intelligence”. McCarthy [53]

“By allowing an organism to see itself as a unified whole, the self-model gave animals a new capacity to interact with the world in an integrated, intelligent manner. The self, is...composed of...body sensations, feelings, autobiographical memory, mental pictures of the body, and a plethora of other self-referential information, which is collated in the brain to compose a conscious self-model....” Thomas Metzinger [79]

Symbolic cognition [3] visualizes the human brain as a Universal Turing Machine, operating on amodal mental representations of the world. These theories are construed out of abstract symbols, devoid of sensorimotor experiences of the physical world and postulate an objective view of the world.

Non-symbolic [3] cognition, on the other hand, promulgates that cognition relies on the experiences of the whole organism – its sense of ‘self’ in its constant interactions with the environment. This leads to grounded [80] mental representations (abstract or real) of information.

Non-symbolic cognition manifests views expressed in the quotes above and Turing’s [1, 81] and inspire the augmented Z-numbers or Z*-numbers defined in this section. These structures draw from the principles of ‘mindfulness’ where understanding affects is the key to social synergy, empathy and contemplative behaviour. Z*-numbers augment Z-number parameters to represent elements defining E in a machine-mind.

A. Definition of Z*-numbers

Given a natural language simple sentence, Y , on a subject X , the ‘Z*-number’ of Y is a 5-tuple $Z_Y^* = < T, C, A, B, AG >$, where, T is the time – a combination of the present, retrospective or prospective moments, C is the context of Y , A is the value of X = the predicate of Y = instantiation (or, restriction on the values) of X given C and T , B is a measure of the reliability or certainty of A given C , X , T and is represented as adjective or adverbial words or phrases, and AG stands for the affect group – a set of ordered triples of (affect, affect-strength, affect-valence [82]) arising out of mind-transactions triggered by instantiation of X given C and T ; multiple ordered triples in an AG for Y represent ‘mixed feelings’ [19] due to ‘mixed emotions’; basic [83] affects are ‘happy’, ‘sad’, ‘fear’, ‘surprise’ and ‘disgust’; affect-strength is typically depicted as adjective or adverbial words or phrases (*e.g.*, very, strongly, etc.); default affect-strength = 0; default affect-valence = positive; affect-valence $\epsilon \{+, -\}$.

A, B and elements of AG are perception-based fuzzy numbers. Some examples of Z*-numbers are:

- (i) $Y_1 = I$ am reading a Wodehouse at the moment.

Therefore, X = What am I reading presently?

$T = \{\text{retrospective, present}\} = \{\text{recent past, now}\}$

$C = \{\text{my current read, personal memories and reviews of Wodehouse's works}\}$

$A = \text{a Wodehouse} \Rightarrow \text{a book by P. G. Wodehouse (an example of 'metonymy')}$

$B = \text{definitely} \leftarrow \text{implied value for certainty in } Y_1$

$AG = \{\text{(happy, very high, +)}\} \leftarrow \text{intuitively, a result of internal processing in the order of:}$

$$\sum \left[\begin{array}{l} \text{Happy personal memories of Wodehouse's works OR} \\ \text{Recall positive reviews of Wodehouse's works by 'attachment figures'} \\ + \text{General mood at } T \end{array} \right], \quad (26.6)$$

where **OR** implies the logical OR operator, and attachment-figures [4, 5] = set of acquaintances who influence judgments.

Thus, $Z_{Y1}^* = <\{\text{recent past, now}\}, \{\text{my current read, personal memories and reviews of Wodehouse's works}\}, \text{a book by P. G. Wodehouse, definitely, } \{\text{(happy, very, +)}\}>$.

(ii) $Y_2 = \text{It takes me about an hour to reach point A!}$

Therefore, $X = \text{time it takes me to reach point A}$

$T = \{\text{retrospective, present}\} = \{\text{over some days, today}\}$

$C = \{\text{average travel time to point A, general road conditions, less travel time today}\}$

$A = \text{an hour}$

$B = \text{about} \Rightarrow \text{usually} \leftarrow \text{explicit value for certainty in } Y_2$

$AG = \{\text{(surprise, very, +), (disgust, high, -)}\} \leftarrow \text{a result of internal processing in the order of:}$

$$\left[\begin{array}{l} \sum (\text{Tedious travel experiences to point 'A'}) \\ + \text{Relatively easier experience today} \end{array} \right] \quad (26.7)$$

Thus, $Z_{Y2}^* = <\{\text{over some days, today}\}, \{\text{average travel time to point A, general road conditions, travel time today}\}, \text{an hour, usually, } \{\text{(surprise, very, +), (disgust, high, -)}\}>$.

Alternatively, if,

$C = \{\text{average travel time to point A, general road conditions, longer travel time today}\}, \text{then}$

$AG = \{\text{(surprise, very, -), (disgust, high, -)}\} \leftarrow \text{a result of internal processing in the order of:}$

$$\left[\begin{array}{l} \sum (\text{Tedious travel experiences to point 'A'}) \\ + \text{Excruciating experience today} \end{array} \right] \quad (26.8)$$

Accordingly, $Z_{Y2}^* = <\{\text{over some days, today}\}, \{\text{average travel time to point A, general road conditions, travel time today}\}, \text{an hour, usually, } \{\text{(surprise, very, -), (disgust, high, -)}\}>$.

The ordered 6-tuple $\langle X, T, C, A, B, AG \rangle$ is referred to as a ‘Z*-valuation’. A set of Z*-valuations for a particular event (E) denotes the ‘Z*-information’ for E and is the stimulus to related decision-making processes. Preliminary rules of Z*-number computations are:

- (i) Precisiation of values of A, B and AG through context-sensitive membership functions, μ_A, μ_B, μ_{AG} respectively, where the context is defined by parameters that capture descriptions of the external world (event under consideration) and internal system conditions.
- (ii) μ_{AG} is based on the polarity and weights of the affects constituting AG . The sub-affects of AG are fuzzy numbers and their strengths are precisiated by membership functions.
- (iii) X and A together define a random event given (T, C) in R , and the probability (p) of this event, may be expressed as:

$$p = \int_R \mu_A(u)p_X(u)d(u), \quad (26.9)$$

where, given (T, C) , u is a real-valued generic value of X and p_X is the underlying (hidden) probability density of X .

- (iv) The Z*-valuation $\langle X, T, C, A, B, AG \rangle$ is viewed as a generalized constraint on X , and is defined by:

Probability (X is A) is $(B^{*C}AG)$, given (T, C)

$$\text{or, } p = \int_R \mu_A(u)p_X(u)d(u) \text{ is } (B^{*C}AG) \quad (26.10)$$

$$\text{i.e., } p = \mu_{B_AG} \int_R \mu_A(u)p_X(u)d(u)$$

$$\text{subject to } \mu_{B_AG} = (B^{*C}AG), \int_R p_X(u)d(u) = 1 \text{ and } (T, C), \quad (26.11)$$

where $(*)^C$ denotes a complex operation agglomerating the mutual [84, 85] roles of belief (B) and emotions (AG) in decision-making. This operator results in the generation of somatic-marker(s) [18, 19] or the system’s ‘feelings’ due to Y . The formalization of $(*)^C$ stresses on the need for careful analyses of the covariance and correlation of beliefs and emotions in everyday decision-making, and is one of our future tasks.

B. Properties

(The following properties of the Z^* -numbers testify our contribution towards extension of the Z-number paradigm. [Refer [86] for other properties and analyses of the Z^* -number paradigm.])

- (i) The Z^* -number parameters of a sentence (Y) are typically instantiated as words or clauses, and these values could either be explicitly mentioned, or implied from the deep-semantics of Y .
- (ii) Z^* -information sets for sentences, for a given C and T , form concept granules.
- (iii) A Z^* -valuation summarizes the objective and subjective percepts of a ‘simple sentence’; while B and AG are thoroughly subjective parameters and X thoroughly objective, A, C and T straddle both perspectives.
- (iv) The Z^* -numbers typically depict ‘active’ ensembles of information in the *working-memory* and *system-memory* constructs of the machine-mind.
- (v) For a Z^* -valuation $< X, T, C, A, B, AG >$ of a sentence Y on an event (E): a) $< T, C >$ define frame-headers activated for E , b) $< X, A >$ symbolize frame-terminal and slot-value respectively, and c) $< B, AG >$ the strength of the terminal-slot_value connectivity, respectively. Z^* -valuations summarize frame-information while Z^* -information summarizes that in a frame-network. Frame-manipulations are typically Z^* -number calculi. Z^* -numbers represent frame-abstractions. Intuitively, the Z^* -numbers leverage between processing sentences (either external inputs to the system or arising endogenously out of reflections on memories) and frame-systems, at real-time.
- (vi) Z^* -numbers of frame-recollections,
“serve as keys to unlock a tome of linguistic knowledge, bathing the brain in the sights, sounds, smells, and physicality of the words’ meaning” [21].
- (vii) Studies in Refs. [15, 17, 87, 88] depict the Time (T) parameter to be crucial to the ‘unified-self’ and ‘reflective-self’ representations within the system-mind. This parameter lends dynamicity to the paradigm.
- (viii) Observations in Refs. [19, 20, 89] justify practical implications of the context (C) in comprehension.

- ly.
- (ix) T and C facilitate anaphora and cataphora resolution.
 - (x) Kahneman's [11] studies, illustrate event-memories as dominated by experiences and consequent feelings – at their peak (for positive events), the nadir (for negative events), and at the very end. The duration is inconsequential, but the affect-strength is the determining factor. This validates the necessity of AG in a Z^* -number.
 - (xi) B and AG arise out of qualia of experiences, anticipations, commonsense and intuition. These parameters act as somatic-markers [19] and index system memories in terms of the affects they evoke. They can be generated online (during current processing) or offline (during reflections or contemplations), and are subject to modifications over time. [Determining the course of action when perceptions undergo major upheavals, causing significant modulations in B and AG of related information, is a sensitive design concern.]
 - (xii) Intuitively, the “degree of comprehension” of a sentence on a subject (X) in a context (C) depends on: a) existing knowledge on $\langle X, C \rangle$, and b) prior granules of comprehension involving $\langle X, C \rangle$.
 - (xiii) The Z^* -number parameters encapsulate: a) *Tropism*: objective organization of the physical world, and b) *Situatedness*: context on which knowledge is retrieved; contexts are subjective subsets of the environment perceived relevant to the agent's goals and actions; may be physical, social, introspective, etc. Tropism and situatedness are essential criteria for emulation of mental-simulation³ – a crucial property of non-symbolic cognition.
 - (xiv) The subjective probability (P_s) of an event (E) is defined by the set of bets about E that an individual (I) is willing to accept. The probability is ‘subjective’ as different individuals may have different probabilities for the same event.
A Z^* -number brings together endogenous mind-elements that form the basis of subjectivity in I for E . These parameters aim representation and precisiation of reasons that define I and P_s .

Deriving from the properties enumerated above, the Z^* -numbers enhance the capability of the Z -numbers. However, emulation of the endogenous arousal of B and AG values – through interactions with the self and others (human users and other artificial cognitive systems) remains a challenge. Other challenges include: a) identification and effective encoding of C , b)

by

activation of sense of T to recollect related system-memories, c) translation of Z^* -numbers of recollections into feelings and qualia in the current perspective, d) definition of affect-operators to integrate Z^* -numbers, e) translation of system sensory inputs into Z^* -numbers or Z^* -information sets, to be processed in conjunction with recollections, results of contemplation, and vestiges of earlier comprehension activities.

A question of importance at this juncture is, the Z^* -numbers embody essential parameters of machine-subjectivity, but are they sufficient?

Enumerating the machine-mind framework and the parameters of subjectivity marks our first step in the realization of a cognitive system. On this note, the article proceeds to detailed illustration of the dynamics of the machine-mind architecture and the role of the Z^* -numbers in the emulation of comprehension of a simple sentence.

26.3.2. Demonstration of the operation of the machine-mind architecture and the Z^* -numbers

Table 26.6 presents an explicit run through the framework – depicting stages of comprehension of a simple natural language sentence, roles of the mind-agencies and the Z^* -numbers of the information being processed. Components in the table abide by the following schematics:

- (i) $\langle \text{bold} \rangle \Rightarrow$ ‘frame_header’; $\langle \text{italics} \rangle \Rightarrow$ ‘terminal_slot’; $\langle \text{bold and italics} \rangle \Rightarrow$ ‘slot_value’
- (ii) $() \Rightarrow$ ‘frame-terminal_slot’ relation
- (iii) Direction of arrow heads denote connectivity destinations – frame-‘terminal_slots’ or ‘slot_values’.
- (iv) Template of Z^* -valuation = $Z^*_{ijk} = \langle \text{Subject } (X), \text{ Time } (T), \text{ Context } (C), \text{ Value } (A), \text{ Confidence } (B), \text{ Affect group } (AG) \rangle$, where $i = \text{time instant or process number}$, $j = \text{thread under } i$, $k = \text{sub-thread under } j$.
- (v) Inference results and data of one stage (stage_i) percolate down to the following stage (stage_j) and are often shared across parallel threads of operation in stage_j .

Our focus being exclusively on portraying processes intrinsic to comprehension, system-activities of **M** have not been highlighted.

Assumptions: Each of the mind-agencies and memory constructs is functional, as per descriptions in Section 26.3.1.1.

Input text: *Smita's eyes were fireflies!*

Expected output: Narrative(s) of comprehension for input.

Table 26.6. Processes of comprehension by the machine-mind framework, using Z^* -numbers to interpret recollected frames, summarize status of comprehension, and analyze the emotional status of the system.

Time	System-action threads	
T ₀	V – Extracts and granulates: ‘Smita’, ‘s’, ‘eyes’, ‘were’, ‘fireflies’, ‘.’	
T ₁	M – Assigns sentence id: S_I	L, AL, ComN and ConN activated for: <Smita>, <s>, <eyes>, <were>, <fireflies>
T ₂	<p>Sy – Activates syntax frames for words in S_I</p> <p>[$Z^*_{21k} = \langle word_k, \{current\}, \{S_I, \text{syntax}\}, A \in \{\text{parts-of-speech}\}, B \in \{\text{definite, uncertain}\}, AG = \text{default} \rangle$, where $word_1 = \langle S \text{ mita} \rangle \rightarrow \langle \text{noun} \rangle$; $word_2 = \langle 's' \rangle \rightarrow \langle \text{is} \rangle \langle \text{was} \rangle \langle \text{has} \rangle$; $word_3 = \langle \text{eyes} \rangle \rightarrow \langle \text{noun} \rangle \langle \text{verb} \rangle$; $word_4 = \langle \text{were} \rangle \rightarrow \langle \text{verb} \rangle$; $word_5 = \langle \text{fireflies} \rangle \rightarrow \langle \text{noun} \rangle \rightarrow \langle \text{common noun} \rangle$]</p>	<p>Re – Recognizes from S_I:</p> <p><Smita> → <proper noun> → <name> → (of) → <human being> → (gender) → <girl></p> <p>[$Z^*_{22} = \langle \text{Smita, \{current, retrospective\}, \{S}_I, who/what is Smita?}, \text{human-female, definite, \{average of affects from experiences on Smita\}} \rangle$]</p>
T ₃	Se – Prunes irrelevant syntax frames for words in S_I : <Smita> → <noun>; <'s'> → <has>; <eyes> → <noun>; <were> → <verb>; <fireflies> → <common noun>	
T ₄	<p>Se – Constructs semantic frames for S_I:</p> <p>Frame1_{S_I} – <Smita> → (what) → <a girl> → <has> → (what) → <eyes></p> <p>[$Z^*_{411} = \langle \text{Smita - human-female possesses, \{current, retrospective\}, \{S}_I, semantics}, \text{eyes, definite, \{average of affects from experiences on Smita and recollections on her eyes\}} \rangle$]</p> <p>Frame2_{S_I} – <eyes> → <were> → (what → <fireflies>)</p> <p>[$Z^*_{412} = \langle \text{Smita's eye description, \{current, retrospective\}, \{S}_I, semantics}, \text{are fireflies, definite, \{average of affects from experiences on Smita and recollections on her eyes, average of affects on recollections of fireflies\}} \rangle$]</p>	<p>Cr – Suggests:</p> <p><eyes> → (possessed by) → <living beings></p> <p>[$Z^*_{421} = \langle \text{what has 'eyes', \{current, retrospective\}, \{S}_I, semantics}, \text{living beings, definite, \{average of recollections of 'eyes' across living things\}} \rangle$]</p> <p><living being> → [human non-human]</p> <p>[$Z^*_{422} = \langle \text{what are 'living beings', \{current, retrospective\}, \{S}_I, semantics}, \text{'human non-human', definite, \{average of recollections of living beings\}} \rangle$]</p> <p><Smita> → <proper noun> → <name> → (of) → <living being></p> <p>[$Z^*_{423} = \langle \text{what is 'Smita', \{current\}, \{S}_I semantics}, \text{a living being, probable, \{average of affects from experiences on Smita - human vs Smita - non-human\}} \rangle$]</p>
T ₅	<p>Su – After series of Z^*-calculi operations and perception-summarization on Z^*-numbers of T₄ for S_I:</p> <p>$Z^*_{51} = \langle \text{what } S_I \text{ means \{current, retrospective\}, \{S}_I, semantics}, \text{Smita - a living being, human-female or non-human - has fireflies for eyes, highly probable, \{average of affects from experiences on Smita - human vs Smita - non-human, recollections of Smita - human-female's eyes, average of affects on recollections of fireflies\}} \rangle$</p> <p>Linguistic sense of S_I: <i>Smita is a living being with eyes that are fireflies. Smita could be a girl or a non-human.</i></p>	

Table 26.6 (Continued)

Time	System-action threads
T ₆	<p>Su – Conjures visual for S_I: <Smita – a girl with fireflies for eyes></p>  <p>[Z^*_6 = <Smita's eye description, {current, retrospective}, $\{S_I$, semantics}, are fireflies, highly uncertain, {(surprise, very high, -), (fear, moderate, +)}>]</p>
T ₇	<p>Su – Senses (from Z^*_6 parameters B = uncertain and AG = {(surprise, high, -), (fear, moderate, +)}) feeling of: Incomprehensibility and incongruity;</p> <p>Su – Initiates re-processing via Cr intervention</p> <p>Sf – Assigns: $\langle\text{system status}\rangle \rightarrow \langle\text{confused, curious}\rangle \rightarrow \langle\text{why}\rangle \rightarrow \langle\text{incomprehensible, incongruity}\rangle$</p>
T ₈	<p>Cr – Activates Sy, Se, Re, Sf, Su to analyze similarities between <eyes> and <fireflies>. Results:</p> <p><eyes> → <properties> → [beautiful bright shine sparkle glow twinkle] <fireflies> → <properties> → [beautiful bright shine glow twinkle]</p> <p>[Z^*_{8I} = <how are 'eyes' and 'fireflies' similar, {current, retrospective}, $\{S_I$, semantics}, 'beautiful bright shine sparkle glow twinkle', highly probably, {(happy, moderate-high, +), (surprise, moderate-high, +)}>]</p>
T ₉	<p>Re – Recalls:</p> <p>1a. <Smita> → <proper noun> → <name of a human being> → <name of a girl></p> <p>1b. <eyes> → (of) → <human beings> → <action> → <bright shine sparkle glow twinkle> → <when> → [very][happy excited]</p> <p>[Z^*_{91} = <when does Smita – a human-girl's – eyes 'shine sparkle glow twinkle', {current, retrospective}, $\{S_I$, semantics}, she very 'happy excited', probably, {(happy, moderate-high, +), (surprise, moderate-high, +)}>]</p> <p>2a. <Smita> → <proper noun> → <name of a non human living being> → <animal></p> <p>2b. <eyes> → (of what) → <animal – cats dogs deer ...> → <action> → <glow> → <when> → <in the dark></p> <p>[Z^*_{922} = <when does Smita – an animal's – eyes 'glow', {current, retrospective}, $\{S_I$, semantics}, in the dark, definite, {(happy, moderate-high, +), (surprise, moderate-high, +), {average affects on recollections of animals with eyes that glow in the dark}}>]</p> <p>Sf – Contemplates:</p> <ul style="list-style-type: none"> • What do 'I', the system, regard as 'beautiful' eyes? • Are my own eyes, V, beautiful? • What do I regard as 'bright shine sparkle glow twinkle' eyes? • When was I very 'happy excited'? • Were my eyes then 'bright shine sparkle glow twinkle'? • ...

Table 26.6 (Continued)

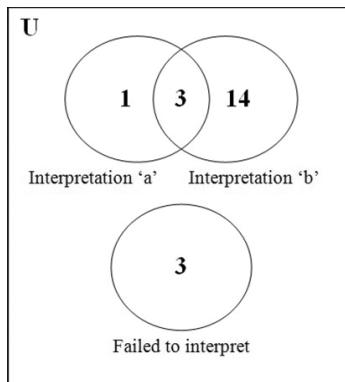
Time	System-action threads
T_{10}	<p>Su – After Z^*-calculi operations and perception-summarization on Z^*-numbers of T_8 and T_9 for S_I:</p> <p>$Z^{*10-11} = \langle \text{what } S_I \text{ means } \{\text{current, retrospective}\}, \{S_I, \text{semantics}\}, \text{Smita – a human-female – has ‘beautiful’ eyes, highly probable, \{average of affects from experiences on Smita and recollections on her eyes, average of affects on recollections of ‘beautiful’ eyes, average of affects on recollections on fireflies\}} \rangle$</p> <p>Linguistic sense₁ of S_I: <i>Smita – a girl – has beautiful eyes.</i></p> <p>$Z^{*10-12} = \langle \text{what } S_I \text{ means } \{\text{current, retrospective}\}, \{S_I, \text{semantics}\}, \text{Smita – a human-female – has ‘bright shining glowing twinkling’ eyes since she is ‘very happy very excited’, highly probable, \{average of affects from experiences on Smita and recollections on her eyes, average of affects on recollections of ‘bright shining glowing twinkling’ eyes, average of affects on ‘happy exciting’ events, average of affects on recollections on fireflies\}} \rangle$</p> <p>Linguistic sense₂ of S_I: <i>Smita – a girl – has ‘bright shining sparkling glowing twinkling’ eyes because she is very ‘happy excited’.</i></p> <p>$Z^{*10-13} = \text{what } S_I \text{ means } \{\text{current, retrospective}\}, \{S_I, \text{semantics}\}, \text{Smita – an animal like ‘cats dogs deer ...’ – has eyes that glow in the dark, probable, \{average affects on recollections of animals with eyes that glow in the dark, average of affects on recollections of fireflies\}} \rangle$</p> <p>Linguistic sense₃ of S_I: <i>Smita – an animal like ‘cat dog deer ...’ – has eyes that glow in the dark.</i></p> <p>Su – Conjures: visuals per interpretation:  <i>< a girl with beautiful eyes ></i>  <i>< a happy girl with bright eyes ></i>  <i>< an animal with eyes that glow in the dark ></i></p> <p>$[Z^{*10-21} = \langle \text{Smita – a girl’s – eye description, \{current, retrospective\}, \{S_I, semantics\}, \{‘beautiful bright shining glowing twinkling’ since ‘happy excited’\}, highly probable, \{(happy, high, +), (surprise, moderate-to-high, +)\}} \rangle]$</p> <p>$Z^{*10-22} = \langle \text{Smita – an animal’s – eye description, \{current, retrospective\}, \{S_I, semantics\}, glowing in the dark, probable, \{(happy, high, +), (surprise, moderate-to-high, +)\}} \rangle$</p>
T_{11}	<p>Se – Identifies: $\langle \text{fireflies} \rangle \rightarrow \langle \text{adjective} \rangle \rightarrow \langle \text{role} \rangle \rightarrow \langle \text{metaphor} \rangle$</p> <p>$[Z^{*11-11} = \langle \text{‘fireflies’ – figure of speech, \{current, retrospective, prospective\}, \{semantics\}, metaphor, highly_probable, \{(happy, moderate-to-high, +), (surprised high-to-moderate, +)\}} \rangle]$</p> <p>Sf – Assigns: $\langle \text{system status} \rangle \rightarrow \langle \text{happy, curious} \rangle \rightarrow \langle \text{why?} \rangle \rightarrow \langle \text{learnt new expression} \rangle$</p> <p>Sf – Contemplates:</p> <ul style="list-style-type: none"> • Does the Smita, I know, have eyes that sparkle like fireflies when she is ‘very happy excited’, so much so that her eyes? • ...
T_{12}	Su – Consolidates: new knowledge on ‘fireflies’ into AL , L and ConN

Observations:

- (i) The example demonstrates comprehension developing incrementally from individual senses (objective and subjective) of words to that for the entire sentence, S_1 . The flow of execution depicted is one of the many interpretation tracks possible.
- (ii) The system engages in fast (e.g. **Re** in T_2 , **Se** in T_4) and slow thinking (e.g. **Cr** in T_4 and T_8 , **Su** in T_6), throughout its operation.
- (iii) **Sf** contemplation in T_9 arises out of social interactions with the real-world.
- (iv) Z^* -valuations activated through the execution have been indicated. These elements represent comprehension status and emotional status (feelings and qualia) of the system-self, and assist mental-simulations or visualizations of interpretations.
- (v) The results (i.e., three senses of S_1) of comprehension were tested for correctness against the thought processes of twenty-one random individuals for S_1 . These individuals were asked to list – over a time of two days – all that their minds processed in relation to S_1 . Synopsis of the survey results are illustrated in Table 26.7 and Figure 26.5. While $sense_1$ and $sense_2$ coincide with majority human interpretations, $sense_3$ is unique to the system; average affects, beliefs and feelings for $sense_1$ and $sense_2$ correspond as well. [We are currently performing more such experiments, to gain insights on thought processes.] Table 26.8 illustrates the machine-mind operations across all the layers of cognition. [Refer to [90] for an illustration of the framework dynamics using Z-numbers].

Table 26.7. Summary of human subject responses to S1.

System Input - Text - 'Smita's eyes were fireflies!'				
Number of human subjects in survey - 21				
No. of survey subjects who stated ...	System interpretation: $sense_1$ only	System interpretation: $sense_2$ only	System interpretation: $sense_1$ and $sense_2$	System interpretation: $sense_3$ only
	1	14	3	0
Average belief on interpretation across test subjects: highly probable - to - definite				
Average affect: surprised; Average feeling: interested				

Fig. 26.5. Venn diagram summarizing human responses to S_1 .Table 26.8. Correspondence between comprehension activity by the machine-mind for S_1 and mind-layer functionality.

Mind-layer	Processing activity
Instinctive	Read given text; updating long-term memory constructs in response to new metaphorical interpretations of “fireflies”.
Learned	Syntax analysis by S_y : Smita → proper noun, 's → [is was has], eyes → [common noun verb], were → verb, fireflies → common noun; Surface semantic analysis by Se of words in S_1 ; Re recalls properties of ‘eyes’, ‘fireflies’, and ‘Smita has eyes → Smita living (human non-human)’
Deliberative	Su simulates visuals of “eyes were fireflies” based on surface interpretation as well as metaphorical sense; Cr activates similarity-analysis(objective and subjective) between ‘fireflies’ and ‘eyes’: [bright glow shining beautiful —...] ⇒ [happy excited ...]; Se prunes irrelevant non-contextual semantic frames, and identifies “fireflies” being used as an adjective or metaphor;
Reflective	Sf awakens affect - incomprehension (confusion); curious (new knowledge gained), and contemplates on what could make Smita [happy excited]? - projection of self onto Smita
Self-reflective	Sf contemplating on what would make the system, [happy excited], what the system considers ‘beautiful’, ...
Self-conscious	Sf contemplating on what would make the system, [happy excited], what the system considers ‘beautiful’, ...

26.3.3. Comparative study

Here we briefly elucidate the conceptual similarities and differences with some contemporary ‘sentient, intelligent, reflective, conscious’ agents – Hearsay [56], CMATTIE [59], IDA [57], LIDA [63] and Genesis project [64]. Our framework draws from the advantages of these systems and aims to augment their abilities.

Similarities – Each of the aforementioned agents and our architecture:

- (i) Is based on the ‘Society of Mind’.
- (ii) Is deliberative and reflective.
- (iii) Relies on co-operative concurrent processing activities across modules for voluntary action selection and constraint satisfaction.
- (iv) Exhibits learning and affects.
- (v) Does not represent ‘forgetfulness’ or mechanisms to handle internal or external distractions.

Differences or distinctive conceptual enhancements in our framework:

- (i) Draws from the ‘Society of Mind’ and the ‘Emotion Machine’ models of cognition. [The Genesis project resembles our framework in this respect.]
- (ii) Acknowledgement of ‘automatic’ and ‘intuitive’ behaviour, and commonsense reasoning. This should predictably prevent activation of the entire complex framework for trivial processing (a key concern in existing systems).
- (iii) Emulation of the machine-‘self’ for ‘self-reflection’ and ‘self-consciousness’ – towards ‘full intelligence’ [53].
 - (a) The Z^* -numbers is our first step in the direction of realization of the machine-self.
- (iv) The concept of encoding ‘reasons’ for failure and success of solution-strategies for possible ‘self-modification’ or ‘self-evolution’ across essential system functions towards system optimization.
- (v) The need for machine-rationalization, through identification and resolution of ‘biased’-decisions and counterfactual-thinking.
- (vi) The notion of ‘self-motivation’ and innate mental rewards prompting ‘curiosity’, learning and solution-trials, despite failures when processing the unknown. This includes the emulation of ‘creativity’ or ‘improvisation’ and the need to ‘stop after a sufficient number of trials’, preventing resource-monopolization by a particularly ‘hard’ problem.

26.4. Conclusion

Real-world comprehension in the human brain is brought about by information flow, integration, and cooperation across functionally differentiated, distributed centers in the brain. These centers collaborate to produce a plethora of conscious experience including perception, emotion, thought and planning, as well as unconscious cognitive and emotional processes.

Drawing from the above and Minsky's theories of cognition, this article elucidates our initial steps towards the design of a cognitive system capable of formulating bespoke interpretations of the real-world for man-machine symbiosis. The macro-components and principle of operation of a framework for a machine-mind have been enumerated, followed by the definition of a new paradigm – the augmented Z-numbers or Z^* -numbers – to encapsulate the objective and subjective elements of real-world information. The machine-mind uses the Z^* -numbers to simulate internal thoughts. The architecture embodies multimodal 'experience' encapsulation towards representation of meanings of sentences and episodic events, endogenous arousal of subjective elements, and artificial creativity. The primary novelty in our design lies in the consideration of the machine-self and its derivatives as crucial to 'true intelligence [53]'.

Design and synthesis of the micro-elements of the framework demands that we understand the physics of innate mind-mechanisms like intentionality, attention-mediation, curiosity, consciousness, qualia, sense of time and space. The answers to these and many more such questions are yet to be unveiled. Thus major discoveries lie ahead before we arrive at a foundation for a computational mind that is anything as basic like the chromosomes, genes and genetic code.

26.5. Acknowledgment

This project is being carried out under the guidance of Professor Sankar K. Pal who is an INAE Chair Professor and a DAE Raja Ramanna Fellow of the Government of India.

References

- [1] A. M. Turing, Computing machinery and intelligence, *Mind*. **49**, 433–460 (1950).
- [2] J. C. R. Licklider, Man-computer symbiosis, *IRE Trans. on Human Factors in Electronics*. **HFE-1**, 4–11 (1960).

- [3] A. Myachykov, C. Scheepers, M. H. Fischer and K. Kessler, Test: A tropic, embodied, and situated theory of cognition, *Topics in Cognitive Science (TopiCS)*. **6**, 442–460 (2014).
- [4] M. Minsky, *The Society of Mind*. Simon & Schuster Inc., NY, USA (1986).
- [5] M. L. Minsky, *The Emotion Machine: Commonsense Thinking, Artificial Intelligence and the Future of the Human Mind*. Simon & Schuster Inc., NY, USA (2006).
- [6] T. A. Harley, *The Psychology of Language: From Data to Theory*, 3rd ed. Psychology Press - Taylor and Francis Group, NY, USA (2008).
- [7] C. J. Price, The anatomy of language: Contributions from functional neuroimaging, *Journal of Anatomy*. **197**, 335–359 (2000).
- [8] L. A. Zadeh, A note on Z-numbers, *Information Sciences*. **181**(14), 2923–2932 (2011).
- [9] P. Langley, J. E. Laird and S. Rogers, Cognitive architectures: Research issues and challenges, *Cognitive Systems Research*. **10**(2), 141–160 (2009).
- [10] P. Singh, Examining the society of mind, *Computing and Informatics*. **22**(6), 521–543 (2003).
- [11] D. Kahneman, *Thinking, Fast and Slow*. Farrar, Straus and Giroux, NY, USA (2011).
- [12] A. Gopnik, A. N. Meltzoff and P. K. Kuhl, *The Scientist in the Crib: What Early Learning Tells us about the Mind*. Harper Collins, NY, USA (1999).
- [13] G. Ryle, *The Concept of Mind*. University of Chicago Press, IL, USA (1949).
- [14] B. J. Baars, *A Cognitive Theory of Consciousness*. Cambridge University Press, Cambridge, UK (1988).
- [15] V. Ramachandran, *The Tell-Tale Brain: A Neuroscientist's Quest for What Makes us Human*. W. W. Norton and Company, NY, USA (2010).
- [16] J. Wiles, Will social robots need to be consciously aware? (in the newsletter of the autonomous mental development technical committee), *IEEE Trans. Autonomous Mental Development*, **11**(2), 14–15 (2014).
- [17] V. Ramachandran, *The Emerging Mind - The Reith Lectures 2003*. Profile Books (in association with BBC), London, UK (2004).
- [18] A. Damasio, *The Feeling of What Happens: Body and Emotion in the Making of Consciousness*. Mariner Books, CA, USA (2000).
- [19] A. Damasio, *Self Comes to Mind: Constructing the Conscious Brain*. Vintage Books, London, UK (2012).
- [20] V. Ramachandran and S. Blakeslee, *Phantoms in the Brain: Probing the Mysteries of the Human Mind*. William Morrow and Company (Harper Collins), NY, USA (1999).
- [21] D. S. Modha, R. Ananthanarayanan, S. K. Esser, A. Ndirango, A. J. Sheridan and R. Singh, Cognitive computing, *Communications of the ACM*, **54**(8), 62–71 (2011).
- [22] E. Winograd, Some observations on prospective remembering. In M. M. Gruneberg, M. E. Morris and R. N. Sykes (Eds). *Remembering That I Did It: Processes and Deficits in Output Monitoring*. (Vol. Practical Aspects of Memory: Current Research and Issues, pp. 348–353). John Wiley and Sons, Ltd, NJ, USA (1988).

- [23] V. Ramachandran and E. Hubbard, Neural cross wiring and synesthesia, *Journal of Vision*. **1**(3) (2001).
- [24] V. Ramachandran and E. Hubbard, The phenomenology of synesthesia, *Journal of Consciousness Studies*. **10**(8), 49–57 (2003).
- [25] M. Conway and C. Pleydell-Pearce, The construction of autobiographical memories in the self-memory system, *Psychological Review*. **107**(2), 261–288 (2000).
- [26] M. D. Rugg and A. P. Yonelinas, Human recognition memory: A cognitive neuroscience perspective, *Trends in Cognitive Sciences*. **7**(7), 313–319 (2003).
- [27] J. L. McGaugh, The amygdala modulates the consolidation of memories of emotionally arousing experiences, *Annual Review of Neuroscience*. **27**, 1–28 (2004).
- [28] O. Hikosaka, Y. Takikawa and R. Kawagoe, Role of the basal ganglia in the control of purposive saccadic eye movements, *Physiological Reviews*, **80**(3), 953–978 (2000).
- [29] A. Stocco, C. Lebiere and J. R. Anderson, Conditional routing of information to the cortex: A model of the basal ganglia's role in cognitive coordination, *Psychological Review*. **117**(2), 541–574 (2010).
- [30] A. Baddeley, The influence of acoustic and semantic similarity on long-term memory for word sequences, *The Quarter Journal of Experimental Psychology*. **18**(4), 302–309 (1966).
- [31] P. Singh, M. Minsky and I. Eslick, Computing commonsense, *BT Technology Journal*. **22**(4), 201–210 (2004).
- [32] P. Singh, A preliminary collection of reflective critics for layered agent architectures. In *Proc. of the Safe Agents Workshop* (2003). [Online]. Available: <http://web.media.mit.edu/~push/ReflectiveCritics.pdf>.
- [33] P. Singh and M. Minsky, An architecture for combining ways to think. In *Proc. International Conference on Integration of Knowledge Intensive Multi-Agent Systems*. 669–674 (2003).
- [34] P. Singh and M. Minsky, An architecture for cognitive diversity. In D. Davis (Ed.). *Visions of Mind: Architectures for Cognition and Affect*. Information Science Publishing, PA, USA 312–331 (2005).
- [35] M. Minsky, A framework for representing knowledge. In P. Winston (Ed.). *The Psychology of Computer Vision*. McGraw-Hill, NY, USA 211–277 (1975).
- [36] D. C. Dennett, *Darwin's Dangerous Idea: Evolution and the Meaning of Life*. Simon & Schuster Inc., NY, USA (1995).
- [37] A. Newen and K. Vogeley, Self-representation: Searching for a neural signature of self-consciousness, *Consciousness and Cognition*. **12**, 529–543 (2003).
- [38] J. M. Mendel, L. A. Zadeh, E. Trillas, R. Yager, J. Lawry, H. Hagras and S. Guadarrama, What computing with words means to me? *IEEE Computational Intelligence Magazine*. **5**(1), 20–26 (2010).
- [39] L. A. Zadeh, Fuzzy logic = Computing with words, *IEEE Trans. Fuzzy Systs.* **4**(2), 103–111 (1996).
- [40] L. A. Zadeh, From computing with numbers to computing with words - from

- manipulation of measurements to manipulation of perceptions, *IEEE Trans. Circuits Systs.* **45**(1), 105–119 (1999)
- [41] L. A. Zadeh, Outline of a new approach to the analysis of complex systems and decision processes, *IEEE Trans. Systs. Man Cyberns.* **3**(1), 28–44 (1973).
 - [42] L. A. Zadeh, Test-score semantics for natural language and meaning representation via pruf. In B. Reiger (Ed.). *Empirical Semantics*. Dr. Broackmeyer University Press, Bochum, Germany. 281–349 (1982).
 - [43] L. A. Zadeh, Fuzzy logic, neural networks and soft computing, *Communications of the ACM.* **37**(3), 77–84 (1994).
 - [44] L. A. Zadeh, Precisiated Natural Language (PNL), *AI Magazine.* **25**(3), 74–91 (1994).
 - [45] L. A. Zadeh, A new direction in AI: Toward a computational theory of perceptions, *AI Magazine.* **22**(1), 73–84 (2001).
 - [46] J. Kacprzyk and S. Zadrożny, Computing with words is an implementable paradigm: Fuzzy queries, linguistic data summaries and natural language generation, *IEEE Trans. Fuzzy Systs.* **18**, 461–472 (2010).
 - [47] R. Kurzweil, *The Singularity is Near*. Viking (Penguin Group), NY, USA (2005).
 - [48] L. A. Zadeh, Intelligent systems revolution: Is it real? (1998). [Online]. Available: <http://www-bisc.cs.berkeley.edu/zadeh/papers>.
 - [49] R. Alieva, A. Alizadeh and O. Huseynovd, The arithmetic of discrete Z-numbers, *Information Sciences.* **290**, 134–155 (2014).
 - [50] R. Banerjee and S. K. Pal, The Z-number enigma: A study through an experiment. In R. R. Yager, A. M. Abbasov, M. R. Reformat and S. N. Shahbazova (Eds.). *Soft Computing: State of the Art Theory and Novel Applications*. (Ser. Studies in fuzziness and soft computing, Vol. 291, pp. 71–88). Springer Berlin Heidelberg (2013).
 - [51] S. K. Pal, R. Banerjee, S. Dutta and S. S. Sarma, An insight into the Z-number approach to CWW, *Fundamenta Informaticae.* **124**(1–2), 197–229 (2013).
 - [52] V. Bush, As we may think, *Atlantic Monthly* (1945). [Online]. Available: <http://www.theatlantic.com/magazine/archive/1945/07/as-we-may-think/303881>.
 - [53] J. McCarthy, The well-designed child, *Artificial Intelligence.* **172**(18), 2003–2014 (2008).
 - [54] G. J. Sussman, A computational model of skill acquisition. Ph.D. dissertation. Massachusetts Institute of Technology, MA, USA (1973).
 - [55] T. Winograd, Procedures as a representation of data in a computer program for understanding natural language. Ph.D. dissertation. Massachusetts Institute of Technology, MA, USA (1971).
 - [56] L. D. Erman, F. Hayes-Roth, V. R. Lesser and D. R. Reddy, The Hearsay-II speech-understanding system: Integrating knowledge to resolve uncertainty, *ACM Computing Surveys.* **12**(2), 213–253 (1980).
 - [57] S. Franklin, IDA: A conscious artifact?, *Journal of Consciousness Studies.* **10**, 47–66 (2003).
 - [58] A. Majumdar, J. Sowa and J. Stewart, Pursuing the goal of language under-

- standing. In *Proc. 16th International Conference on Conceptual Structures: Knowledge Visualization and Reasoning*, 21–42. Springer Berlin Heidelberg (2008).
- [59] L. McCauley, S. Franklin and M. Bogner, An emotion-based “conscious” software agent architecture. In A. Paiva (Ed.). *Affective Interactions* (Ser. Lecture Notes on Artificial Intelligence, Vol. 1814, pp. 107–120). Springer Berlin Heidelberg (2000).
 - [60] B. Morgan, A substrate for accountable layered systems. Ph.D. dissertation. Massachusetts Institute of Technology, MA, USA (2013).
 - [61] P. Singh, Em-one: An architecture for reflective commonsense thinking. Ph.D. dissertation. Massachusetts Institute of Technology, MA, USA (2005).
 - [62] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, N. Schlaefer and C. Welty, Building Watson: An overview of the DeepQA project , *AI Magazine*. **31**(3), 59–78 (2010).
 - [63] S. Franklin, T. Madl, S. D’Mello and J. Snairder, LIDA: A systems-level architecture for cognition, emotion, and learning, *IEEE Trans. Autonomous Mental Development*. **6**(1), 19–41 (2014).
 - [64] P. H. Winston, The Genesis story understanding and story telling system: A 21st century step toward Artificial Intelligence. *White paper on story understanding* (2014). [Online]. Available: <http://groups.csail.mit.edu/genesis/papers/-StoryWhitePaper.pdf>.
 - [65] E. B. Baum, Project to build programs that understand. In B. Goertzel, P. Hitzler and M. Hutter (Eds.), *Proc. Second Conference on Artificial General Intelligence* (Ser. Advances in Intelligent Systems Research, Vol. 8, pp. 1–6). Atlantis Press, Paris, France (2009).
 - [66] B. J. Grosz, What question would Turing pose today?, *AI Magazine*. **33**(4), 73–81 (2012).
 - [67] R. Banerjee and S. K. Pal, Text comprehension and the computational mind agencies, *Natural Computing*, Springer. **14**(4), 603–635 (2015).
 - [68] D. Ariely, *Predictably Irrational: The Hidden Forces that Shape our Decisions*. Harper Collins, NY, USA (2008).
 - [69] S. K. Pal and R. Banerjee, Context-granulation and subjective information quantification, *Theoretical Computer Science*. **448**, 2–14 (2013).
 - [70] A. Tversky and D. Kahneman, Judgment under uncertainty: Heuristics and biases, *Science (New Series)*. **185**(4157), 1124–1131 (1974).
 - [71] S. K. Robinson and L. Aronica, *Finding Your Element: How to Discover Your Talents and Passions and Transform Your Life*. Viking (Penguin Group), NY, USA (2013).
 - [72] O. Sacks, *The Man who Mistook his Wife for a Hat*. Picador, London, UK (1986).
 - [73] A. K. Seth, E. Izhikevich, G. N. Reeke and G. M. Edelman, Theories and measures of consciousness: An extended framework, *Proc. of the National Academy of Sciences (PNAS)*. **103**(28), 10799–10804 (2006).
 - [74] C. Havasi, R. Speer and J. Alonso, Conceptnet 3: A flexible, multilingual semantic network for common sense knowledge. In *Proc.*

- Recent Advances in Natural Language Processing. [Online]. Available: <http://web.media.mit.edu/~jalonso/cnet3.pdf>.
- [75] B. J. Baars, *In the Theater of Consciousness: The Workspace of the Mind*. Oxford University Press, Oxford, UK (1997).
 - [76] B. J. Baars, The conscious access hypothesis: Origins and recent evidence, *Trends in Cognitive Sciences*. **6**(1), 47–52 (2002).
 - [77] S. Pinker, *How the Mind Works*. W. W. Norton & Company, NY, USA (1997).
 - [78] J. Backus, Can programming be liberated from the von Neumann style? A functional style and its algebra of programs (ACM Turing Award Lecture), *Communications of the ACM*. **21**(8), 613–641 (1978).
 - [79] T. Metzinger, The phantom that controls you? How the self-model works, *Being Human*(2013). [Online]. Available: <http://www.beinghuman.org/-article/phantom-controls-you>.
 - [80] L. W. Barsalou, Grounded cognition: Past, present, and future, *Topics in Cognitive Science (TopiCS)*, **2**, 716–724 (2010).
 - [81] A. Turing, Intelligent machinery (1949). [Online]. Available: <http://www.alanturing.net/turing.archive/archive/l/132/L32-011.html>.
 - [82] W. G. Lehnert, Plot units and narrative summarization, *Cognitive Science*. **4**, 293–331 (1981).
 - [83] P. Ekman, Basic emotions. In T. Dalgleish and M. Power (Eds.), *Handbook of Cognition and Emotion*, 45–60. John Wiley and Sons, Ltd, Sussex, UK (1999).
 - [84] N. Frijda, A. Manstead and S. Bem, The influence of emotions on beliefs, In N. Frijda, A. Manstead and S. Bem (Eds.), *Emotions and Beliefs: How Feelings Influence Thoughts*, 1–9. Cambridge University Press, Cambridge, UK (2000).
 - [85] M. P. Paulus and A. J. Yu, Emotion and decision-making: Affect-driven belief systems in anxiety and depression, *Trends in Cognitive Sciences*, **16**(9), 476–483 (2012).
 - [86] R. Banerjee and S. K. Pal, Z*-numbers: Augmented Z-numbers for machine-subjectivity representation, *Information Sciences*, **323**, 143–178 (2015).
 - [87] M. Pieck, P. Weiss, K. Zilles, H. Markowitch and G. Fink, Differential remoteness and emotional tone modulate the neural correlates of autobiographical memory, *Brain*. **126**, 650–668 (2003).
 - [88] D. Falk, *In Search of Time: The History, Physics, and Philosophy of Time*. St. Martin's Griffin, London, UK (2010).
 - [89] S. C. Furtak, O. J. Ahmed and R. D. Burwell, Single neuron activity and theta modulation in postrhinal cortex during visual object discrimination, *Neuron*. **76**(5), 976–988 (2012).
 - [90] R. Banerjee and S. K. Pal, On Z-numbers and the machine-mind for natural language comprehension. In D. E. Tamir, N. D. Rishe and A. Kandel (Eds.), *Fifty Years of Fuzzy Logic and its Applications*, (Ser. Studies in Fuzziness and Soft Computing, Vol. 326, pp. 415–457). Springer (2015).

Author Index

- Acharya, A., 75 Hirata-Jr., R., 545
- Bahrampour, S., 199 Jankowski, A., 323
- Banerjee, R., 805 Jayadeva, 395
- Basak, J., 135 Joshi, V., 789
- Bovolo, F., 713 Kothari, R., 789
- Bruzzone, L., 713 Lovell, B. C., 605
- Cesar-Jr., R. M., 545 Maji, P., 513
- Chakraborty, A., 767 Manwani, N., 161
- Chakravarthy, V., 789 Mooney, R. J., 75
- Chandra, S., 395 Morimitsu, H., 545
- Chandra Sekhar, C., 407 Murty, K. S. R., 633
- Chaudhuri, S., 735 Nasrabadi, N. M., 199
- Chaudhury, S., 581 Nguyen, H. S., 323
- Chellappa, R., 113
- Deb, K., 293 Pal, A., 1
- Demir, B., 713 Pal, S. K., 1, 487, 805
- Deravi, F., 255 Palguna, D., 789
- Dey, L., 581 Patel, V. M., 113
- Dileep, A. D., 407 Paul, S., 513
- Ganguly, N., 767 Pedrycz, W., 217
- Ganivada, A., 487 Pizzi, N. J., 217
- Gherman, B. G., 255
- Ghosh, A., 687 Ray, A., 199
- Ghosh, J., 75 Ray, S. S., 487
- Ghosh, S., 687, 767 Reddy, P. R., 633
- Girolami, M., 37 Rudra, K., 767
- Hashimoto, M., 545 Sastry, P. S., 167
- Hassan, E., 581

- Sharma, M. U., 735
Sirlantzis, K., 255
Skowron, A., 323
Soman, S., 395
Stathopoulos, V., 37
Subramaniam, L. V., 789
Subudhi, B. N., 687
Sunil, T. T., 735
Veena, T., 407
Verma, I., 581
Wiliem, A., 605
Yegnanarayana, B., 633
Zhang, B., 663

Subject Index

- ϵ -constraint method, 298
- λ -kNN classifier, 151
- 0-1 loss function, 187
- abstraction, 12
- accuracy, 17
- accuracy of approximation, 331
- active knowledge transfer, 81
- active learning, 75, 715
- adaptive judgment, 355, 369
- adaptive ridge regression, 148
- adjacency matrix, 556
- Akaike information criterion, 137
- Alignment kernels, 439
- ANA, 607
- Anti-Nuclear Antibody (ANA), 606
- appearance descriptor, 548
- approximate reasoning methods, 369
- approximation spaces, 325, 338
- approximation to the total risk, 400
- Artificial Neural Networks (ANNs),
 - 14, 488
- ASM, 640
- attribute noise, 171
- audio search, 633
- automatic relevance determination,
 - 42
- average performance of STD, 646
- average precision, 646
- average similarity matrix, 752
- B-indiscernible objects, 329
- B-vectors, 147, 152
- back propagation algorithm, 491
- backpropagation, 261
- Bag of Riemannian Words (BoRW),
 - 620
- bag of words approach, 611
- basic granule (atom) of knowledge,
 - 324
- basis functions, 38
- basis pursuit, 115
- Bat call identification, 60
- Baum-Welch algorithm, 642
- Bayes classifier, 5, 140
- Bayes information criterion, 137
- bespoke real-world comprehension,
 - 805
- between class variance, 750
- bias variance trade off, 743
- bias-variance, 140
- Big data, 25
- binary probit, 51
- blur, 570
- (approximate) boolean reasoning, 325
- borderline cases, 324
- boundary region, 324
- Box vectors, 147
- brain modeling, 487
- Bresenham algorithm, 551
- brute force search, 744
- business intelligence, 591
- C++, 562
- CAD systems, 606
- Cauchy kernel, 155

- cavity distribution, 46, 47, 51
Cell Pyramid Matching (CPM), 616
centroid, 547
change of scale, 572
change or motion detection, 688
channel mismatch, 647
Chebyshev distance, 560
Chernoff bound, 793
class conditional label noise, 171
class imbalance, 114
classification, 256, 487
classification accuracy, 737
classification algorithms (classifiers), 332
classification filtering, 175
classification trees, 8
classifier, 256
cluster analysis, 513
clustering, 20, 487
clustering of target responses, 742
co-expressed gene clustering, 522
co-expressed micro RNA clustering, 528
Codebook-based IMK, 433
color features, 666
combination of generative and discriminative models, 642
combinatorial optimization, 745
combined system, 656
common sense reasoning, 368
community detection, 767, 769, 785, 786
complex game discovery task, 370
complex granules (c-granules), 355, 357
complexity measures, 264
computational theory of perceptions (CTP), 13
computing with words, 13
computing with words and perception, 366
conditional sentiment, 791
conditionally Gaussian, 651
conditioned topic learning, 586
consistent (deterministic) decision table, 333
constant partition label noise, 171
content-based image retrieval, 407
context-based clustering, 243, 244
continuous action-set learning automata, 192
core, 335
Covariance Descriptors (CovDs), 618
covariance function, 41
covariate shift, 114
crisp (precise) set, 324
crowded tournament selection operator, 304
crowding distance, 304
crowding distance assignment procedure, 305
Darwinian survival, 487
data capture and signal processing section of the E-Nose, 738
data complexity, 259
data indexing, 633
data mining, 24, 633
data set bias, 114
DBM, 639
DC programming, 192
decision fusion, 263
decision tree, 138
deductive logic, 368
definable sets, 329
degree of the dependency, 335
Delaunay triangulation, 561, 568
dependency factors, 489
descriptor, 548
descriptor computation, 573
descriptor extraction, 551
desirable feature of a good gas sensor, 749
detection and classification of target gases, 735
detector, 547
Diabetes Mellitus (DM), 663
dictionary learning, 116, 200, 207
dimensionality reduction, 248
direct index matching, 640
directed graph, 547
discernibility matrix, 348

- discernibility relation, 346
Discrete Cosine Transform (DCT), 612
discrete Fourier transform, 551
discrete or continuous, 637
dissimilarity function, 558
dissimilarity in target responses, 749
distribution capturing capability of GBRBM, 652
distribution of energy, 743
DM, 664, 665, 670–673, 677–681, 684
DM-sans NPDR, 665, 677, 680–683
DNA computing, 488
domain adaptation, 713
domain adaptation with low-rank reconstruction, 124
domain adaptive dictionary learning, 119
dominant sentiment, 791
DTW, 634, 639
Dual Region (DR), 616
dynamic kernels, 407
dynamic reducts, 353
dynamic time alignment kernel, 441
dynamic time warping kernel, 440
dynamic time warping, DTW, 60, 62
- E-Nose sensor response, 740
edge, 547
edit distance alignment, 640
electronic noses, 735
Elitist Non-Dominated Sorting GA (NSGA-II), 302
Emotion Machine, 806
empirical error, 400
empirical risk, 188, 398
empirical risk functional, 136
empirical risk minimization, 395
energy function for GBRBM, 651
ensemble learning method, 601
entropy measurements, 265
EP algorithm, 44, 45, 49, 51
EP approximation, 57, 58
EP axrpoximation, 48
estimating the joint probability density, 640
- Euclidean distance, 564
evaluate the classification performance by cross validation, 737
event-based news analytics, 582
evolutionary algorithm, 488, 489
evolutionary multi-criterion optimization (EMO), 294
Expectation Maximization (EM), 648
expectation propagation, 44, 45
exponential map, 619
- face recognition, 125
fast computation, 552
fat margin, 402
fat margin classifier, 398
feasible set, 295
feature, 547
feature extraction, 4
feature selection, 4, 736, 742
feature selection for machine learning, 737
feature-level fusion, 200
feed-forward neural networks, 255
FFTW library, 562
filter based method, 745
filter methods, 737
Fisher kernel, 427, 437
Fourier descriptor, 564
Frâchet or Riemannian mean, 621
fractal geometry, 488
full forward probability, 640
fully connected networks, 260
fuzzy clustering, 20, 514
fuzzy decision classes, 488
fuzzy information granulation, 488
fuzzy linear classifiers, 226
fuzzy logic, 488
fuzzy Petri net, 234, 235, 237, 238, 240
fuzzy rough granular neural network, 487
fuzzy rough set, 22, 487
fuzzy rough-fuzzy set, 22
fuzzy self organizing map, 489
fuzzy sets, 11, 218, 220
fuzzy sets and fuzzy rough sets, 487

- fuzzy tolerance relation, 495
- gap tolerant classifier, 398, 402
- gap tolerant hyperplane classifier, 399
- gas detection, 742
- gas detection accuracy, 736
- gas signature, 736
- Gaussian Cloud, 267
- Gaussian + Exponential kernel, 156
- Gaussian mixture model (GMM), 407
- Gaussian mixture model based IMK, 435
- Gaussian parity problem, 153
- Gaussian posterior vector, 650
- Gaussian Process Prior, 40
- Gaussian Processes, 39
- GRBM, 634
- GRBM posteriors, 656, 658
- generalization, 12, 262
- generalization error, 136
- generalization performance, 399
- generalized approximation space, 338
- generalized decision class, 334
- generalized domain adaptive dictionary learning, 120
- generalized rough sets, 23
- Genetic Algorithms (GAs), 15
- geodesic distance, 619
- geometry features, 664
- Global alignment kernel, 441
- global descriptor, 548
- GMM, 634, 640
- GMM posteriors, 658
- GMM posteriors are more robust, 650
- GMM supervector kernel, 430
- GMM-UBM mean interval kernel, 430
- GP classifiers, 37
- gradient decent method, 489
- gradualness of knowledge, 354
- granular computing, 223, 224, 232, 487
- Granular Computing (GrC), 19, 326, 356
- granular hyperbox-driven classifiers, 232
- granular neural networks, 489
- granularity of knowledge, 354
- granulation and perception, 487
- granule, 487
- graph, 547
- group behaviour, 487
- growth of the sensing quality function, 761
- hard_suit, 357
- hashtag classification, 769, 770
- heat-map based visualizations, 598
- HEP-2 cell classification problem, 609
- Hessian-Affine, 564
- heterogeneous domain adaptation, 118
- hidden Markov model (HMM), 407
- hierarchical mixture of experts, 139
- Hierarchical Sparse Representation-Based Domain Adaptation, 123
- high bias/low variance classifiers, 743
- high resource languages, 638
- histogram encoding methods, 613
- HMM forced alignment, 644
- HMM-based intermediate matching kernel, 442
- HMM-MLP, 634
- HMM-MLP hybrid model, 656
- HMMs, 639
- idioms, 767–771, 774–777, 779, 782, 784–786
- image annotation, 407
- image classification, 407
- image gradient, 547
- image matching, 407
- image time series, 713
- imbalanced classification, 396
- imperfect knowledge, 323
- IN-Vocabulary (INV) words, 638
- inconsistent (non-deterministic) decision table, 333
- indiscernibility relation, 329
- inducing approximate reasoning schemes, 342
- inductive extensions of approximation spaces, 343

- inductive logic, 368
inductive reasoning, 342
information granulation, 356
information granule, 222, 244
information system, 328, 329
innovation, 315
intelligent-systems, 806
Interactive (Rough) Granular Computing (I(R)GrC), 326, 355
interactive computations, 357
interest point, 547
intermediate matching kernel (IMK), 407
isomorphism, 547, 549

joint density, 652
joint density capturing, 648

k-nearest neighbor(s), 138, 595
k-nearest neighbor (*k*-NN) rules, 6
Karcher mean, 621
kernel based matching, 408
kernel function, 415
Kernel methods, 408
kernel trick, 8
keygraph, 549
keygraph correspondence, 554
keygraph detection, 550
keypoint, 547
keypoint detection, 573
keypoint framework, 549
keytuple, 556
keyword spotting, 633
Knowledge Discovery in Databases (KDD), 24

L-risk, 187, 189
L2 Regularisation, 39
label noise, 167, 168
labelled speech data, 642
Lagrangian multiplier, 143
land-cover maps updating, 713
language and pronunciation, 634
large deviation theory, 793
LASSO (least absolute shrinkage and selection operator), 149

lazy evaluation scheme, 758
learning algorithms, 256
Learning Vector Quantization (LVQ), 21
Least Square Kernel Machine with Box Constraints (LSKMBC), 147
Length-Constrained Minimum Average (LCMA), 641
Levenberg-Marquardt algorithm, 562
likelihood ratio kernel, 439
linear classifiers, 225
linear discriminant analysis (LDA), 7
linear separability, 225
linguistic terms, 493
link function, 37, 38
link_suit, 357
local descriptor, 548
local feature vectors (FVs), 407, 410
local thresholding, 699
logarithm map, 619
logic canvas, 250
logic processor, 230, 231
loss function, 136, 187
low bias/high variance classifiers, 743
low dimensionality, 552
low resource languages, 648
low-cost training sets, 713
low-rank decomposition, 117
lower and the upper approximation, 324
lower approximation, 17, 329
LVCSR, 637

machine-self, 805
man-machine symbiosis, 805
manually labelled data, 658
MAP estimate, 696
margin, 138
match the GMM posteriograms, 650
matching, 635
matching based kernel, 431
matching kernel, 432
matroids, 757
maximization of submodular functions, 758

- maximization of submodular functions over a uniform matroid, 755
- maximum margin classifiers, 8
- maximum margin hyperplane, 415
- measure of dissimilarity, 749
- meta-measurements, 264
- MFCCs, 635
- MFD, 548
- Minimal Complexity Machine (MCM), 399
- minimum description length principle, 136, 332, 336
- minimum similarity, 749
- miss rate of longer query words is less, 656
- mitosis stage, 608
- mixture of experts, 264
- MLP, 639
- MLP posteriors, 636
- mobile phone, 568
- model keygraph, 550
- model non-stationary signals, 642
- modelling correlated data, 651
- modified k -nearest neighbor, 150
- modify the shape of the warping path, 647
- modular artificial neural network, 258
- monotonic label noise, 171
- Monte Carlo, 801
- motion estimation, 688
- MSER, 548, 565, 568
- multi-class classification, 396
- Multi-criterion decision-making (MCDM), 293
- Multi-criterion Optimization (MO), 293
- multi-layer compound Markov Random Field model, 694
- multi-layer perceptron, 491
- multi-source domain adaptation, 118
- multilayer feed-forward networks, 260
- Multilayer Perceptron (MLP), 643
- multimodal content retrieval, 585
- multimodal data mining, 581
- multimodal news analytics, 582
- multimodal retrieval, 586
- multinomial probit model, 43, 51, 58
- multiple classifier system, 264
- multitask learning, 75
- Naive Bayes classifier, 190
- named-entities, 593
- natural robustness, 552
- nearest neighbor (NN) classifier, 227
- network information criterion, 137
- neural networks, 139, 229
- node, 547
- non-parallel hyperplanes, 396
- Non-proliferative Diabetic Retinopathy (NPDR), 664
- non-uniform label noise, 171
- normalization, 552
- NPDR, 670–673, 677, 680–684
- NPDR/DM-sans NPDR (DM without NPDR), 664, 684
- object recognition, 126
- Occam's Razor, 135, 136
- occlusion, 572
- one-against-all strategy, 145
- online social network, 767, 769
- ontology approximation, 341, 362
- OOV, 638
- OpenCV, 562, 568
- optimal sensor set, 749
- optimal subset, 737
- Oracle, 286
- Pareto-optimal solutions, 296
- pattern matching, 255
- pattern recognition, 3, 219, 220, 229, 235, 325
- patterns, 487
- Perception Based Computing (PBC), 365
- perception based information, 488
- perspective, 572
- Petri net, 234, 237
- phonetic posteriors, 639, 658
- PIR, 209
- PN, 634

- polynomial boundary, 270
pose estimation, 558
positive region, 334
posterior representation, 634
predictor function, 37, 38
prime implicants, 346
principal component analysis, 742
principle of parsimony, 135, 143
probabilistic finite-state automata, 200
probabilistic fitting procedure, 558
probabilistic sequence kernel, 428
probabilistic topic models, 77
probability product kernel, 439
probit function, 47
Probit model, 43
problem-solving, 18

quadratic loss, 148
quantum computing and granular computing, 488

ramp loss, 190
Random Forest(s), 10, 601
rank function is a submodular in nature, 760
RANSAC, 558, 567
reducing the variance, 135
reduct, 18, 335
regularization using model complexity, 142
remote sensing, 713
representation techniques, 658
response characteristics to different target gases, 736
Restricted Boltzmann Machine (RBM), 651
ridge regression framework, 148
risk functional, 135
risk minimization, 187
RLSC reproducing Hilbert kernels, 149
robust PCA, 117
robust representation, 635
robust to speaker variability, 644
rough (imprecise, vague) set, 324

rough (inexact) set, 331
rough inclusion relation, 337
rough logic, 488
rough membership function, 336
rough mereology, 340
rough set (RS) approach, 323
rough set based methods for inducing classifiers or clusters, 342
rough sets, 16
rough-fuzzy c-means, 516
rough-fuzzy clustering, 515
rough-fuzzy granulation, 355
rough-fuzzy set, 22
roughness, 17

sample selection bias, 114
sampling, 793
satisfiability relation, 370
SAW resonators, 739
SAW sensors, 739
scalability, 574
Scale Invariant Feature Transform (SIFT), 612
scene correspondence, 573
scene image, 412
scene keygraph, 550
score space kernels, 436
segmental DTW, 640
segmentation of brain MR image, 530
seismic, 210
selection based on individual performance, 745
selection of optimal sensor combination, 753
selection through redundancy elimination, 746
self-evaluation, 487
self-organization, 487
Self-organizing Feature Maps (SOFMs), 21
self-repetition, 487
self-reproduction, 487
semi-supervised domain adaptation, 118
semi-supervised learning, 715
semiotic c-granules, 359

- sensing quality function, 756
 sensitivity and selectivity, 736
 sensor array, 738
 sensor selection, 744
 sensor selection as a combinatorial optimization problem, 749
 sensor selection based on individual sensor performance, 747
 sensor selection through redundancy elimination, 748
 sensor selection using filter methods, 748
 sensor selection using maximization of submodular functions over uniform matroids, 758
 sensor selection using submodular function maximization, 755
 sensor selection using wrapper method, 745
 separability index, 750
 sequences of local feature vectors, 407
 sequential pattern, 410
 set of local feature vectors (FVs), 407, 411
 SIFT, 547, 563, 569
 sigmoid loss, 190
 signal energy, 743
 signature of object, 329
 significant eigenvalues, 743
 similarity between sensors, 752
 smoothing, 646
 social robot, 806
 society of mind, 806
 Soft Assignment (SA), 613
 soft computing, 21, 487, 488
 soft-indicator output, 138
 soft_suit, 357
 Softmax model, 43
 sorites paradoxes, 328
 Sparse Coding (SC), 614
 sparse representation, 115, 200, 203
 spatial data, 584
 Spatial Pyramid Matching (SPM), 615
 spatio-temporal, 692
 spatio-temporal object detection, 688
 speaker identification, 407
 speaker-invariant nature of posterior features, 646
 speaker-invariant representation, 634
 spectrograms, 635
 speech emotion recognition, 407
 speech-specific, 637
 spoken term detection, 633
 squared exponential covariance, 42
 state of the hidden units, 654
 statistical pattern recognition, 5
 statistical properties of data, 648
 STD, 633
 string covariance function, 43
 structural risk, 398
 structural risk minimization, 136, 395
 sub systems in a practical E-Nose, 736
 submodular set functions, 756
 subsequence DTW, 640, 650, 658
 subsequence matching, 443
 subspace interpolation via dictionary learning, 122
 subword LVCSR, 638
 suitability of a sensor subset, 737
 summation kernel, 431
 superfluity in the outcome, 137, 138
 supervised classification, 713
 supervised learning, 256
 supervised methods, 5
 supervised or unsupervised, 637
 supervised pattern recognition, 3
 supervised topic models, 79
 supervised topics, 78
 support vector machine(s), 408, 743
 Support Vector Machines (SVMs), 8
 SURF, 547, 569
 surface acoustic wave, 736
 symbolic dynamic filtering, 200, 201
 symmetric KL divergence, 655
 symmetric loss function, 190, 193
 synthetic datasets, 266
 target classification, 209
 target separability, 749, 750
 task decomposition, 263
 telephone monitoring, 633

- Telugu, 658
temporal segmentation, 688
texture, 664
the connectionist approach, 14
the fuzzy set theoretic approach, 11
the hybrid approach, 21
the rough set theoretic approach, 16
the syntactic approach, 10
thermal equilibrium, 652
thinking machines, 805
thresholding, 558
Tikhonov regularization, 136, 143
tilt, 570
tilted distribution, 46, 47, 51, 52, 57
tongue color, 664, 666–669, 684
tongue color feature, 665, 669
tongue color feature vector, 669, 670
tongue geometry feature, 665
tongue texture feature, 665
total risk minimization, 395
total variance distance, 277
training, 736
training and validation, 744
transfer learning, 76, 715
triangular fuzzy number, 226
triangular global alignment kernel, 442
twin support vector machine, 395
twin SVM Applications, 398
twin SVM extensions, 397
twin SVM Formulation, 396
Twitter, 767–770, 774, 775, 777–779, 782–786
Twitter trends, 768, 774

unbalanced datasets, 396
unconstrained linear regression, 144

uniform label noise, 171
universal background model, 428
unlabelled speech, 658
unsupervised domain adaptation, 118
unsupervised pattern discovery, 641
unsupervised pattern recognition, 4
upper approximation, 17, 329
upper bound on the VC dimension, 400

vague (imprecise) concepts, 327
vague concept, 323, 324
validation, 736
Vapnik-Chervonenkis (VC) dimension, 137
variance learning, 654
varying length patterns, 407
VC dimension, 399
vector correlation measurement, 266
Vector Quantization (VQ), 613, 638
vertice, 547
visual domain adaptation, 118
voice dialling, 633

WCN, 637
weaker discrimination, 553
weight adaptation, 273
well separated gas signatures, 749
WER, 637
WisTech program, 326
within class variance, 750
wrapper method, 737, 745

 Z^* -numbers, 13
 Z -number(s), 13, 805
zero centered, 401

This page intentionally left blank

About the Editors



Sankar K. Pal (www.isical.ac.in/~sankar) is a *Distinguished Scientist* and former *Director* of the Indian Statistical Institute. He is also a J. C. Bose Fellow and Raja Ramanna Fellow of the Government of India. He founded the *Machine Intelligence Unit* and the *Center for Soft Computing Research: A National Facility* in the Institute at its Kolkata campus. He had received a Ph.D. in Radio Physics and Electronics from the University of Calcutta in

1979, and another Ph.D. in Electrical Engineering along with DIC from the Imperial College, University of London, in 1982.

Prof. Pal worked at the University of California, Berkeley and the University of Maryland, College Park during 1986–87; the NASA Johnson Space Center, Houston, Texas during 1990–92 and 1994; and in the US Naval Research Laboratory, Washington DC in 2004. Since 1997, he has been a *Distinguished Visitor* of the IEEE Computer Society, for the Asia-Pacific Region, and has held visiting positions in several universities in Italy, Poland, Hong Kong and Australia.

He is a co-author of seventeen books and more than four hundred research publications in the areas of Pattern Recognition and Machine Learning, Image Processing, Data Mining, Web Intelligence, Soft Computing, Neural Nets, Genetic Algorithms, Fuzzy Sets, Rough Sets Social Network Analysis and Bioinformatics. He has visited about forty countries as a Keynote/ Invited speaker.

He is a *Fellow* of the IEEE, the Academy of Sciences for the Developing World (TWAS), International Association for Pattern Recognition, International Association of Fuzzy Systems, and all the four National Academies for Science & Engineering in India. He serves/ has served on the editorial boards of twenty-two international journals including several IEEE Transactions.

He has received the S. S. Bhatnagar Prize (the most coveted award for a scientist in India) in 1990, the Padma Shri (one of the highest civilian awards) in 2013 from the President of India and many prestigious awards in India and abroad including the G. D. Birla Award (1999), the Jawaharlal Nehru Fellowship (1993), the Khwarizmi International Award (2000) from the President of Iran, NASA Tech Brief Award (USA, 1993), the Outstanding Paper Award (1994) from the IEEE Transactions on Neural Networks, and Indian Science Congress-P. C. Mahalanobis Birth Centenary Gold Medal (2005–06) from the Prime Minister of India for Lifetime Achievement.



Amita Pal (nee Pathak) obtained a B.Sc. (Honours) degree in Statistics from the Presidency College, Calcutta in 1979, an M.Sc. degree in Statistics from the University of Calcutta in 1981, and a Ph.D. degree in Statistics from the Indian Statistical Institute, Calcutta in 1991. She visited the Imperial College of Science, Technology and Medicine, London, in 1994 on a UNDP fellowship. Upon her return, she joined the Institute as a Lecturer in 1994, and is currently working as an associate professor in the Interdisciplinary Statistical Research Unit of the same institute. Her research interests are mainly in the areas of pattern recognition and image processing.